UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
MARSHALL DIVISION

---

OPTi Inc.,
Plaintiff

v. Civil Action No. 2-04CV-377

nVidia Corporation,
Defendant

---

# EXPERT REPORT OF ALAN JAY SMITH IN RESPONSE TO THE INVALIDITY EXPERT REPORT OF ROBERT COLWELL REGARDING U.S. PATENT NO. 5,710,906

Submitted May 31, 2006

Confidential

## 1. INTRODUCTION

My name is Alan Jay Smith, and I have been retained by OPTi Inc. ("OPTi") as an expert witness in the matter of OPTi v. nVidia. I have been asked to prepare this expert report. This report provides my opinion on whether OPTi's Patent No. 5,710,906 ("the '906 Patent") is valid, in view of the Expert Report of R. Colwell, Ph.D.

My qualifications and compensation were previously described in my Expert Report on the Infringement of the '906 Patent. A copy of my resume is attached as Exhibit A.

## 2. SUMMARY OF OPINIONS

Based on the analysis that appears in this report, it is my opinion that asserted claims 1, 7, 8, 21, 26 of the '906 Patent are valid, despite the opinions presented in the Expert Report of R. Colwell, Ph.D.

Specifically, neither Intel nor Compaq can show conception prior to OPTi. The other references (Amini, IBM TDB, Motorola) do not disclose the same invention, and in the case of Motorola, conception cannot be shown to preceed that by OPTi. None of these latter renders the '906 Patent obvious.

There are a number of objective indications of non-obviousness: commercial success, others tried and failed to make the invention, taking of licenses under the patent by others, and long felt need.

The '906 Patent does provide support by written description.

In addition to my testimony regarding validity, I may be asked to provide a tutorial and/or a historical overview of the relevant technology, to discuss the state of the art, and may be asked to respond to testimony and evidence offered by defendant nVidia Corporation.

## 3. LEGAL PRINCIPLES RELIED UPON

I have been instructed and/or understand as follows:

### 3.1. Claim Construction and Interpretation

For the construction of the language of claims at issue I have relied on the Court's Markman Opinion of April 24, 2006, as discussed more fully in my infringement report dated May 16, 2006.

### 3.2. Ordinary Skill In The Art

My understanding is that the claims of a patent are to be read and understood as if by one of ordinary skill in the art, "skill in the art" referring to a level of expertise in the subject matter of the patent. I consider the art relevant to the '906 patent to be that of the design and implementation of digital systems. One of ordinary skill in the art would typically have a B.S. or M.S. in electrical engineering, computer science or computer engineering, and would have 2-5 years of experience designing and building digital systems, above and beyond any study related to the degree.

### 3.3. Anticipation

My understanding of anticipation is as follows: In order for a patent to be anticipated, each and every limitation of the claim must be present within a single item of prior art, whether that prior art is a publication, a prior patent, a prior invention, a prior public use or sale, or some other item of prior art. A printed publication or patent will not be an anticipation unless it contains a description of the invention covered by the patent claims that is sufficiently detailed to teach a skilled person how to make and use the invention without undue experimentation.

### 3.4. Obviousness

My understanding of obviousness is as follows: A patent is not valid if the invention would have been obvious to a person of ordinary skill in the field at the time the invention was made. Obviousness may be shown by considering more than one item of prior art. In considering whether to combine items of prior art, there must be some motivation or suggestion to a skilled person to make the combination covered by the patent claims.

Among the objective indications of non-obviousness are: (a) commercial success of products covered by the patent claims; (b) failed attempts by others to make the invention; (c) the taking of licenses under the patent by others; and (d) long felt need.

### 3.5. First Invention

My understanding is that someone would be the first inventor if s/he was the first to conceive (even if not the first to reduce to practice) if s/he exercised reasonable diligence in reducing the invention to practice, and that the invention was not abandoned, suppressed or concealed.

#### 3.5.1. Conception

I have been given the following instruction and case law regarding conception:

*Burroughs Wellcome Co. v Barr Laboratories, Inc.*, 40 F.3d 1223 (Fed. Cir. 1994):

> "Conception is the touchstone of inventorship, the completion of the mental part of invention." *Id.* at 1228.

> Conception is "the formation in the mind of the inventor, of a definite and permanent idea of the complete and operative invention, as it is hereafter to be applied in practice." *Id.*
>
> "An idea is definite and permanent when the inventor has a specific, settled idea, a particular solution to the problem at hand, not just a general goal or research plan he hopes to pursue." *Id.*
>
> "Conception is complete only when the idea is so clearly defined in the inventor's mind that only ordinary skill would be necessary to reduce the invention to practice, without extensive research or experimentation." *Id.*
>
> "The conception analysis necessarily turns on the inventor's ability to describe his invention with particularity. Until he can do so, he cannot prove possession of the complete mental picture of the invention." *Id.*

*Oka v. Youssefyeh*, 849 F.2d 581 (Fed. Cir. 1988):

> "Conception requires (1) the idea of the structure of the [invention], and (2) possession of an operative method of making it." *Id.* at 582.
>
> "Conception may conveniently be considered as consisting of two parts. the first part is 'the directing conception' and may be defined as the idea or conception that a certain desired result may be obtained by following a particular general plan...The second part of conception is 'the selection of the means for effectively carrying out the directing conception." *Id.* at 583.
>
> Conception has not occurred if the inventor "lack[s] ...possession of a method for making [his invention]." *Id.* at 584.

*Slip Track Systems, Inc. v. Metal-Lite, Inc.*, 304 F.3d 1256 (Fed. Cir. 2002):

> "Conception must include every feature or limitation of the claimed invention." *Id.* at 1263.

I interpret the above legal instruction to require that functional aspects of the invention must have been part of the conception. In the case of complex asynchronous devices such as chipsets, which interface external busses with internal buses, there are many special cases (referred to as "corner cases" by Thome (deposition transcript p. 78)) that need to be specified and designed for. (*See* below.) Until *all* significant special cases have been recognized and incorporated in the conception, the conception is not complete.

> 0078
>
> Q. This bug that you identified, is this what you would sometimes describe as a "corner case"?

A. Yeah. So we would say it was an unusual, you know, sequence, and it took an awful lot of effort to even -- to get it to occur. And so it was one that was -- you know, you could -- you could describe as a corner case, although it was certainly important to have it fixed before we took it to production because, otherwise, you would -- you know, even -- even though it only happened infrequently, infrequently having the wrong data is not a good thing for customer applications.

0111

we were running it and when we found a -- a particular bug.

The way we would do it is, we'd sit down with the PCIFACTs tool, and first we would run through every different kind of command. Then we would run through -- then we'd let it run -- iterate through all the different variances of that command.

So it would like run, you know, a cycle at offset one byte in the cache and offset two bytes in the cache, offset three bytes. And so we'd kind of iterate through all those combinations to make sure that they all worked properly. And so there were thousands upon thousands of different possible combinations. And we would use that software and verify the operation of the -- of the design that way.

And then the meat grinder software that I -- I spoke of earlier also used the same hardware cards to generate all the different kinds of PCI bus cycles, all the different PCI traffic. And that tool would be one where we would then let them all run simultaneously and you'd generate a huge amount of fairly random traffic just to try to find some of the kind of the -- the corner case bugs, primarily.

### 3.5.2. Corroboration of Conception

*Slip Track Systems, Inc. v. Metal-Lite, Inc.*, 304 F.3d 1256 (Fed. Cir. 2002):

"Inventor testimony alone is insufficient to prove conception; some form of corroboration must be shown." *Id.* at 1263 .

*In re Jolley*, 308 F. 3d 1317 (Fed. Cir. 2002):

"Because conception is a mental act, it must be proven by evidence showing what the inventor has disclosed to others and what that disclosure means to one of ordinary skill in the art. *Id.* at 1321.

*Coleman v. Dines*, 754 F.2d 353 (Fed. Cir. 1985):

"Conception must be proved by corroborating evidence which shows that the inventor disclosed to others his completed thoughts expressed in such clear terms as to enable those skilled in the art to make the invention." *Id.* at 359.

*Price v. Symsek*, 988 F.2d 1187 (Fed. Cir. 1993):

"[C]onception by an inventor, for the purpose of establishing priority, can not be proved by his mere allegation nor by his unsupported testimony where there has been no disclosure to others or embodiment of the invention in some clearly perceptible form, such as drawings or model, with sufficient proof of identity in point of time. For otherwise[,] such facile means of establishing priority of invention would, in many cases, offer great temptation to perjury, and would have the effect of virtually precluding the adverse party from the possibility of rebutting such evidence. Hence it has been ruled in many cases that the mere unsupported evidence of the alleged inventor, on an issue of priority, as to conception and the time thereof, can not be received as sufficient proof of prior conception." Id., at 1195 (*quoting* Mergenthaler v. Scudder, 11 A.D.C. 264 (D.C. Cir. 1897)).

*Trovan, Ltd. v. Sokymat SA, Irori*, 299 F.3d 1292 (Fed. Cir. 2002):

"Reliable evidence of corroboration preferably comes in the form of physical records that were made contemporaneously with the alleged prior invention." *Id.* at 1302-03.

"Circumstantial evidence about the inventive process may also corroborate... Additionally, oral testimony of someone other than the alleged inventor may corroborate." *Id.* at 1303.

*Medichem, S.A. v. Rolabo, S.L.*, 437 F.3d 1157 (Fed. Cir. 2006):

As to the corroborative value of inventor's notes --- "As far as the corroborative value of the inventors' notebooks is concerned, they were not witnessed, and they do not provide an 'independent' source of authority on the issue of reduction to practice. Hence, they have minimum corroborative value." *Id.* at 1172.

*Ethicon, Inc. v. U.S. Surgical Corp.*, 135 F.3d 1456 (Fed. Cir. 1998):

> "Under the 'rule of reason' standard for corroborating evidence, the trial court must consider corroborating evidence in context, make necessary credibility determinations, and assign appropriate probative weight to the evidence." *Id.* at 1464.

### 3.5.3. Reduction to Practice and its relation to Conception

My understanding of reduction to practice is: "An invention is reduced to practice either when a patent application is filed or when the invention is made and shown to work for its intended purpose."

In implementing an electronic device, or for that matter almost any complex device, electronic, mechanical or otherwise, there is almost always a "debugging" phase. Debugging generally refers to eliminating errors in the design and implementation.

For purposes of reduction to practice, I separate errors in design from those of implementation. An error of design is when some significant aspect of the design (including "corner cases", as mentioned above) have not been recognized or accounted for. An error of implementation is when the person of ordinary skill implementing the conception simply made a mistake, even though the conception is complete (covers all aspects and cases) and error free.

If there are any significant errors of design, then the conception is not complete until they are identified and eliminated.

### 3.5.4. Due Diligence in Reducing Invention to Practice

*Brown v. Barbacid*, 436 F.3d 1376 (Fed. Cir. 2006):

> "The basic inquiry is whether, on all of the evidence, there was reasonably continuing activity to reduce the invention to practice. There is no rule requiring a specific kind of activity in determining whether the applicant was reasonably diligent in proceeding toward an actual or constructive reduction to practice." *Id.* at 1380.
>
> The purpose of the reasonable diligence requirement "is to assure that the invention was not abandoned or unreasonably delayed by the first inventor during the period after the second inventor entered the field." *Id.* at 1379.

*Griffith v. Kanamaru*, 816 F.2d 624 (Fed. Cir. 1987):

"Delays in reduction to practice caused by an inventor's efforts to refine an invention to the most marketable and profitable form have not been accepted as sufficient excuses for inactivity." *Id.* at 627.

"A review of caselaw on excuses for inactivity in reduction to practice reveals a common thread that courts may consider the reasonable everyday problems and limitations encountered by an inventor." *Id.*, at 626.

*Beidler v. Caps*, 17 C.C.P.A. 703 (C.C.P.A.1929):

Delay in reducing an invention to practice for "mere business considerations" does not constitute the requisite diligence. *Id.* at 710.

*Thompson v. Dunn*, 35 C.C.P.A. 957 (C.C.P.A. 1948):

"It is well settled, however, that diligence will not wait upon commercial expediency." *Id.* at 963.

*Naber v. Cricchi*, 567 F.2d 382, 385 (Fed. Cir. 1977):

Diligence can be shown by a variety of activities, however, it is "well settled that, to satisfy the 'reasonable diligence' requirement of 35 U.S.C. 102(g), the work relied on must ordinarily be directly related to reduction to practice of the invention of the counts in issue." *Id.* at 385.

*Brunswick Corp. v. United States*, 34 Fed. Cl. 532 (Fed Cl. 1995):

"Continuous diligence covers constructing or testing an invention or embodiment as well as preparing a patent prosecution or any similar activity directed to reducing the invention to practice. Any lapse in diligence, including premature commercial exploitation may prove fatal to establishing an earlier priority date." *Id.* at 590-91.

### 3.5.5. Testing Issues

*Scott v. Finney*, 34 F.3d 1058 (Fed. Cir. 1994):

To show reduction to practice an inventor "must demonstrate that the invention is suitable for its intended purpose." *Id.* at 1061.

Testing of the product is often necessary to show proof of actual reduction to practice - frequently with "[c]omplex inventions and problems." When this is the case "the embodiment relied on as evidence of priority must actually work for its intended purpose." *Id.* at 1062.

## 4. <u>Materials Considered</u>

In forming the opinions presented here, I have relied on the following materials as well as additional materials cited throughout the body of this Report:

United States Patent No. 5,710,906

PCI Local Bus Specification, Revision 2.0, April 30, 1993

Memorandum Opinion and Order, April 24, 2006, Document 96, Case 2-04-cv-00377-TJW. (Markman Order)

Expert Report of R. Colwell, Ph.D., May 16, 2006.

Deposition transcript and exhibits for the deposition of George Hayek, April 27, 2006.

Deposition transcript and exhibits of Barry Davis, April 20, 2006

Deposition transcript and exhibits for the deposition of Michael Collins, May 3, 2006.

Deposition transcript and exhibits for the deposition of Christopher Bryant, April 17, 2006.

Deposition transcript and exhibits for the deposition of Nader Amini, May 8, 2006.

Deposition transcript and exhibits for the deposition of Michael Garcia, March 31, 2006.

Deposition transcript and exhibits for the deposition of Brian Langendorf, May 4, 2006.

Deposition transcript and exhibits for the deposition of Ali Oztaskin, April 28, 2006.

Deposition transcript and exhibits for the deposition of Brian Reynolds, March 30, 2006.

Deposition transcript and exhibits for the deposition of Gary Thome, May 10, 2006.

Deposition transcript and exhibits for the deposition of Bruce Young, April 22, 2006.

Deposition transcript and exhibits for the deposition of Subir Ghosh, January 6, 2006.

Deposition transcript and exhibits for the deposition of Tom Shanley, May 16, 2006.

U.S. Patent No. 5,581,714, December 3, 1996, Amini et al.

U.S. Patent No. 5,634,073, May 27, 1996, Collins et al.

U.S. Patent No. 5,630,094, May 13, 1997, Hayek et al.

IBM Technical Disclosure Bulletin, 36, 10, October, 1993, 187-191.

Tom Shanley, "PCI System Architecture," Mindshare Press, 1993.

Michael Garcia, Brian Reynolds, "Single Chip PCI Bridge and Memory Controller for PowerPC Microprocessors," Proc. ICCD, October 10-12, 1994, 409-412.

Japanese Patent Office, Official Gazette for Unexamined Patent Applications, Application H5-193313, Filed August 4, 1993.

Karl Wang et al., "A PCI Bridge Chip (MPC105) with a Cache and Memory Controller for PowerPC Microprocessor," Proc. HotChips Conference, August, 1994.

James Nicholson and Fred Strietelmeier, "New Micro Channel Features," Proc. IEEE Compcon, February 28-March 2, 1990, 179-182.

Declaration of Subir Ghosh in Support of Request for Further Consideration of DS Documents D2-1 Through D2-7.

Patent application 09/631,564, filed August 2, 2000.

Tien-fu Chen, "Data Prefetching for High Performance Processors", Ph.D. Thesis, 1993, U. Washington.

Lee, Roland Lun, "The Effectiveness of Caches and Data Prefetch Buffers in Large-Scale Shared Memory Multiprocessors", Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1987.

Frederik Dahlgren, Michel Dubois and Per Stenstrom, "Sequential Hardware Prefetching in Shared-Memory Multiprocessors", IEEETPDS, 6, 7, July, 1995, 773-746.

Dean Tullsen and Susan Eggers, "Limitations of Cache Prefetching on a Bus-Based Multiprocessor," Proc. ISCA 1993, 278-288.

William Chen, Scott Mahlke, Pohuo Chang and Wen-mei Hwu, "Data Access Microarchitectures for Superscalar Processors with Compiler-Assisted Data Prefetching", Proc. MICRO-24, November, 1991, 69-73.

Alan J Smith, "Cache Memories", ACM Computing Surveys, September, 1982, 473-530.

Jean-Loup Baer and Tien-fu Chen, "An Effective On-Chip Preloading Scheme To Reduce Data Access Penalty", Proc. Supercomputing '91, November, 1991.

Tien-fu Chen and Jean-Loup Baer, "A Performance Study of Software and Hardware Data Prefetching Schemes," Proc. ISCA 21, April, 1994.

Roland Lee, Pen-chung Yew, Duncan Lawrie, "Data Prefetching in Shared Memory Multiprocessor," Proc. IC, August, 1987, 28-31.

Edward Gornish, Elana Granston, Alexander Veidenbaum, "Compiler-directed Data Prefetching in Multiprocessors with Memory Hierarchies," Proc. ICS, June, 1990, 354-368.

In addition, I have relied on my education, training and experience in the relevant technology, my knowledge of the relevant literature and may rely on technical encyclopedias and dictionaries if necessary. I may also rely on any further evidence to be produced in this case.

## 5. Technology Background

The relevant technology background for the '906 patent was previously provided in my Expert Report on Infringement. In this section, I repeat portions of that material, and elaborate on points especially relevant to validity.

For most computers, the rate at which the instruction and execution units operate exceed the speed at which their corresponding main memories operate. So most computers (including the ones envisioned in the patent at issue) use "cache memories," ('906 Patent, 1:48-4:46), which are small, high speed memories used to temporarily hold those portions of the contents of main memory which are (believed to be) currently in use; blocks ("lines") of information are loaded into memory either when they are first used or sometimes they are "prefetched" (loaded in advance, in the expectation that it will be used). Cache memories operate at speeds comparable to the instruction and execution units. There may be more than one level of cache memory (referred to as "level 1 cache", or "L1 cache", "level 2 cache", (L2 cache), etc.). The highest level(s) of the cache memories are typically considered to be part of the CPU (central processing unit), referred to in figures 1 and 2 as the "host processing subsystem." Additional levels of cache may be associated with main memory. (*See* Smith, "Cache Memories", Computing Surveys, 14, 3, September, 1982, 473-530, for a general discussion of cache memories).

Computer systems also typically have additional levels of the memory hierarchy, consisting of disk drives, tape drives, etc. Computer systems may also have other devices, such as modems, audio cards, ethernet cards, etc., as discussed below, which may be connected to the system by a PCI bus. Information is frequently transferred between the devices attached to the PCI bus and main memory. Such transfers use main memory addresses (rather than cache locations to specify the memory locations being read or written). Such transfers also frequently involve the transfers of blocks of information, consisting of a sequential transfer of data to or from sequential addresses in memory.

The computer systems of interest are internally connected with "buses", which are sets of wires (signal lines) used to transmit information between various units. A given type of bus is designed according to, and operates according to, a set of rules, as given in the appropriate specification. (E.g. the PCI bus ('906 Patent, 4:66-5:16) is defined in the PCI Local Bus Specification, Revision 2.0.) Such specifications indicate the number of wires, the types of signals, the timing of the signals, the voltages and currents required, etc.

As noted above, cache memories are used to temporarily hold portions of the information normally residing in main memory. Information in caches is typically stored in "blocks" or "lines", which are sets of contiguous bytes of information. (Contiguous bytes are those that are stored at a set of sequential addresses, e.g. 16, 17, 18, 19, ...31). When the I/E units modify information, that modification may be transmitted ("written") immediately to main memory ("write through" or "store-through"), or may be written initially to the cache memory, and later copied back ("copy-back" or "writeback") to main memory. The technology of interest concerns copy-back caches. ('906 Patent, 2:48-4:32). Copyback or writeback caches are preferred in high performance systems because their use usually leads to higher performance of the system as compared to a system implemented with a write-through cache.

The problem with a copy-back cache memory is that the information cached in the cache memory may be different than that stored in main memory. But it is essential for correctness that all references to a given piece of information reference the most recent (and correct) copy. When the I/E units reference information, they first attempt to find the information in the cache memory, and they read and write it there. Conversely, references to that information from an external device attached to the PCI bus are to the copy in main memory. If the copy of the data in cache memory has been updated and is more current than that in the main memory, then allowing a direct reference to the main memory copy will yield incorrect results. ('906 Patent, 5:17-6:4). That problem is called the "cache consistency" or "cache coherency" problem. There are various schemes for maintaining consistency between the contents of cache memory and the contents of main memory. (*See* Sweazey and Smith ("A Class of Compatible Cache Coherency Protocols and Their Support by the IEEE Futurebus", Proc. 13'th Ann. Int. Symp. on Computer Architecture, Tokyo, Japan, June, 1986, . 414-423) for a discussion of how this can be done on a bus called the "Futurebus).

The prior art computer systems described in the patent at issue ('906 Patent, 5:17-6:4) solve this problem as follows: prior to a read or write to main memory by an external device attached to the PCI device, an inquiry signal is sent by the device across the PCI bus and through the SYSC/IPC chip to the host processing system. The inquiry will include the main memory address that is being accessed. This inquiry is called a "snoop." The Host Processing Subsystem will determine whether the cache memory contains the information being addressed. If it does contain the information, and the cache memory copy is more current than the copy in main memory, then main memory will be updated. (In general, on a read from memory, the cache copy may be either maintained or discarded. On a write to memory, the cache copy may either be updated or discarded.) The least efficient implementation of this process will make an inquiry for every 4 bytes (the width of the PCI bus). A better implementation will make an inquiry for every cache line ('906 Patent, 5:25-31). But in either case, there is a delay for each inquiry - in the prior art, a line would be transferred, and then a delay would occur for an inquiry cycle before the next line could be transferred to/from main memory. The patent specifically says ('906 Patent, 4:36-46):

> In Pentium-based systems, a bus master initiates an inquire cycle by driving the inquire address onto the CPU address leads and asserting EADS#. The processor responds by asserting its HIT# output if the specified data line is present in the L1 cache. The processor also asserts an HITM# output if the specified L1 cache line is in the M (modified) state. Thus, HITM#, when asserted, indicates that the L1 cache contains a more current copy of the data than is in main memory. The processor then automatically conducts a write-back cycle while the external bus master waits. By this process, therefore, the external bus master will be able to access the desired line in main memory without any further concern that the processor's L1 cache contains a more current copy of the data.

Transfers across the PCI bus are frequently sequential. By "sequential," we mean that data is transferred (read or written) in sequential order to sequentially numbered locations in main memory. (Data moves across the PCI bus 4-bytes at a time, and thus each transfer contains the next 4 bytes.) The patent specifically states ('906 Patent, 5:5-16):

> The PCI bus achieves very high performance, in part because its basic data transfer mode is by burst. That is, data is always transferred to or from a PCI device in a known sequence of data units defined by a known sequence of data unit addresses in an address space. In the "cache line" burst mode, exactly four transfers take place. In the "linear" burst mode, any number of transfers (including 1) can take place to/from linearly sequential addresses until either the initiator or the target terminates the transaction. In either mode, the initiator need only specify the starting address because both parties know the sequence of addresses which follow.

The problem is that burst transfers that cross cache line boundaries incur a delay for each inquiry cycle ('906 Patent, 5:41-51, 5:57-6:4):

> Even with inquire cycles limited to one per cache line, a problem still exists if the desired burst length proceeds past a cache line boundary. Conventional chipsets determine when a new access in the burst is in a new cache line, and they withhold the PCI-bus TRDY# signal while they perform the necessary inquire cycle for the new cache line. If the Pentium processor asserts HITM#, then the chipset stops the PCI-bus transaction (using a target disconnect termination), allows the L1 cache to perform a write-back operation, and resumes with a new inquire when the PCI master restarts the transaction where it left off."... "If the inquire cycle for the new line of cache does not produce HITM#, then there is no need to stop the PCI transaction. Instead, conventional chipsets

> merely withhold TRDY# for the time required to perform the inquire cycle, and then assert TRDY# when the inquire cycle has completed without HITM#.
>
> The time required to perform the inquire cycle, however, is significant. On the PCI-bus, a delay of eight PCI-bus clock cycles may be incurred each time that a linear burst transaction crosses a cache line boundary. A definite need, therefore, exists for a mechanism which allows PCI-bus bursts to proceed past a cache line boundary whenever possible. Such a mechanism can help PCI-bus masters achieve the full promise of high-speed data transfers afforded by the PCI-bus burst transfer protocol.

So the idea of the patent is that when there is a sequential burst transfer, to issue an inquiry to the next sequential line (line K) concurrently with the current transfer (line K-1), so that by the time the transfer of the K'th line is ready to begin, the inquiry has already been completed, and if no writeback (copyback) is needed, little or no delay will be experienced between the transfer of line K-1 and line K. The result is that in the case of a read from main memory, to data which is not in a modified state in cache memory, the flow of bytes from the main memory to the bus master (the unit that initiated the read request) will be uninterrupted. In the other case, of a write to main memory, the flow of bytes to main memory will similarly be uninterrupted provided that the cache memory does not contain a copy of the data stored in the locations being written to. The patent says: ('906 Patent, 6:8-24):

> According to the invention, roughly described, when a PCI-bus controller receives a request from a PCI-bus master to transfer data with an address in secondary memory, the controller performs an initial inquire cycle and withholds TRDY# to the PCI-bus master until any write-back cycle completes. The controller then allows the burst access to take place between secondary memory and the PCI-bus master, and simultaneously and predictively, performs an inquire cycle of the L1 cache for the next cache line. In this manner, if the PCI burst does in fact continue past the cache line boundary, the new inquire cycle will already have taken place (or will already be in progress), thereby allowing the burst to proceed with at most a short delay absent a hit-modified condition. This avoids the need to incur the penalty of stopping the transfer on the PCI bus and restarting it anew at a later time, every time a linear burst transaction crosses a cache line boundary.

It is important to observe what the patent describes. Consider the case of a read from main memory. Absent the scheme of the patent, a line would be read from main memory, then an inquiry would take place (and a writeback (copyback) would occur if necessary), and then the next line would be read from main memory. What the patent describes is a scheme by which bytes are read from main memory in a steady flow, without any need for interruptions for inquiry cycles. Nothing in the patent specification requires that there be only one line in transit between the main memory and the bus master or that there be no buffering in the course of that transfer. A "transfer" from memory to a PCI bus master may be direct, or may be indirect through various additional chips which may or may not contain buffering. This is much like a conveyor belt - as long as the belt is loaded fully and continuously at the source, items will flow steadily and at maximum capacity, independent of the length or routing of the belt. Conversely, if the belt is operated with a protocol in which the belt unloading person asks for more items only when he has unloaded the belt, then the full capacity of the belt will not be used. This is clear from the following wording of the '906 Patent:

> In order to minimize or eliminate delays at cache line boundaries, as previously described, the system controller 116 performs a predictive snoop ("pre-snoop") of the second cache line address of the burst, prior to completion of the last PCI-bus data transfer from the initial cache line address of the burst ... the pre-snoop takes place simultaneously with at least one data transfer taking place on the PCI-bus 118. The predictive snoop is "predictive" because it is performed even though the system controller 116 does not yet know whether the PCI device 138 desires to continue the burst beyond the cache line boundary." ('906 Patent, 14:36-49).
>
> Thus, the inquiry cycle for the second cache line has been completed before the last data transfer takes place in the first cache line. Assuming the first transfer does in fact proceed beyond the cache line boundary, the first data transfer (Dword 20) of the second line of data can take place without stopping the burst and without inserting any additional PCI-bus wait states (see arrow 442)." ('906 Patent, 14:60-67).

The improvement in performance contemplated by this technique can be quantified. In the embodiment disclosed, the chipset transfers one dword of data every two PCI clock cycles. Thus it would take 16 clock cycles to transfer a line of data. If multiple lines are transferred, it would still take 16 clock cycles to transfer each line. In contrast, according to the patent, (5:57-6:4) in the prior art systems, such multi-line transfers would, in addition, require at least eight clock cycles to disconnect and reconnect each time a cache line was crossed. Thus, each line would take a minimum of 24 clock cycles, or 50% longer than the disclosed embodiment. The patent contemplated that this performance improvement might well be considerably greater. It notes that the two cycles per dword implementation was not necessary to the invention, and that other embodiments could dispense with this timing feature. ('906 Patent, 14:10-13). Under these circumstances, each line transfer employing the patented method would take eight clock cycles, as opposed to 16 clock cycles for the prior art method and the performance enhancement would be 100%. [redacted]

## 6. OPTi - Conception and Reduction to Practice

The '906 Patent is assigned to OPTi, plaintiff in this case. With regard to other claimants to the same invention, it is necessary to show that OPTi was the first to conceive of the invention, that it was diligent in reducing the invention to practice, and that it did not abandon, conceal or suppress the invention. In this section, I address the issue of Conception and Reduction to Practice.

The named inventors on the '906 Patent are Subir Ghosh and Hsu-Tien Tung. Exhibit 41 to the Ghosh deposition consists of schematics for the OPTi Viper product, which was a realization of the '906 invention.

> Q. Can you explain how you conceived the invention described in the '906 patent?
>
> A. How I came up with the idea?
>
> Q. Sure.
>
> A. Well, as I mentioned earlier, I was responsible for designing architecting and also architecting the chipset, as well as for the microarchitecture. When we started working on our -- this particular chipset, which was called Viper at that moment, that was probably the first time PCI was introduced to the OPTi chipsets.
>
> 36
>
> Q. The Viper chipset practiced the predictive snooping of your '906 patent; correct?
>
> A. That is correct.
>
> 167

Exhibit 41 to the Ghosh deposition (OPTINVIDIA 008334-46) shows some schematics for the Viper product. OPTINVIDIA 008339 shows the logic in the upper half of figure 9 of the '906 specification, and is dated May 30, 1994. OPTINVIDIA 008340 includes the logic in the lower half of figure 9, and is dated 6/15/94. In construing Claim 21, which calls, among other things for "sequentially transferring data units" across a cache line boundary (and in construing Claim 26, which calls for doing so at a constant rate), the court has determined that the structure which is the "means for initiating a next line inquiry" is the logic shown in figure 9 of the patent. OPTINVIDIA 024971 of exhibit 40 to the Ghosh deposition shows simulated timing diagrams for the pre-snoop function of the '906 Patent. OPTINVIDIA 024956 discusses the pre-snoop. OPTINVIDIA 024909 of Ex. 40 is dated 8/28/94; a couple of other pages have earlier dates. I believe that OPTINVIDIA 024971 and 024956 are likely to be earlier than OPTINVIDIA 024909. Given that chips were ordered on 8/23/94 (see below), the simulated timing diagrams are likely to be much earlier than 8/28/94.

OPTINVIDIA 007442 shows a purchase order to TSMC to make Viper chips and is dated 8/23/94 (project 557). A memo dated November 12, 1994 (OPTINVIDIA 008332) indicates that as of that date, pre-snoop was enabled and working, i.e. the invention was made and shown to work for its intended purpose. OPTINVIDIA 007687 shows OPTi shipments to customers for the 557 (Viper) part as early as 1/11/95.

The timing of the above events indicates to me that the conception of the pre-snoop invention was complete no later than June 15, 1994, and that OPTi showed diligence in reducing the invention to practice. The invention was not abandoned, suppressed or concealed.

## 7. <u>The Motorola MPC 105 Did Not Embody The Claims At Issue And Is Not Prior Art</u>

In the Expert Report of R. Colwell, Ph.D., it is alleged that the asserted claims of the '906 Patent are invalid in view of the prior invention of Motorola employees, including Michael Garcia, Christopher Bryant and Brian Reynolds.

In this section, I show why that is not so. First, the Motorola invention, whatever it may be, is not the invention of the asserted claims of the '906 Patent. Second, the record does not support the assertion that Motorola had fully conceived of its invention prior to June, 1994.

### 7.1. The MPC 105 Did Not Embody The Claims At Issue

The Motorola MPC105 bridge chip had a speculative snoop feature (Brian Reynolds, 7:6-11). In the MPC105, the transfer stopped after each line, even though there was a pre-snoop. Brian Reynolds describes that feature as follows:

0112

Q. Okay. Now, do you recall that bursts on the MPC105 and 106 were of a finite length; in other words, they couldn't go on indefinitely?

A. Yes.

Q. They were limited to 32-bits, right?

A. I believe -- 32 bytes.

Q. 32 bytes. Which is, say, one line?

A. One cache line, yes.

Q. One cache line. So on the MPC105 and 106, when you had a PCI transaction, it was, by definition, one cache line in length; is that correct?

A. Or less.

Q. Or less. Okay. So on the MPC105 and on the MPC106 -- well, strike that.

And --

0113

MR. BRODY: Let's mark this. Let's mark this as Exhibit 11.

(Plaintiff's Exhibit No. 11 marked)

Q. (By Mr. Brody) I have asked the court reporter to mark as -- I'm sorry -- Plaintiff's Deposition Exhibit 11 a one-page document with Bates number OPTINVIDIA 026361. Do you have that in front of you, sir?

A. I do.

Q. Okay. This is taken from a Motorola web site, and it involves a question and answer about the MPC106. Do you see that?

A. Yes.

Q. Okay. And the person who is writing into the web site says we have a question about the 106. It says when we try to transfer data with memory read line versus memory read multiple line requests an address phase followed by eight data transfers occurs, I guess, and then there is a target abort. And basically they want to understand what's haening. Do you see that?

A. Uh-huh, yes.

Q. And then the help desk folks at Motorola write back and they say, "The 106 as a target Supports the PCI memory-read, memory-read-line, and memory-read-multiple commands. That is, it will

0114

respond to these transactions with DEVSEL."

DEVSEL is a signal for indicating that a device has been selected, right?

A. Correct. It indicates that you as a target are going to be the one providing the requested data, or accepting in the case of a write transaction.

Q. Okay. "However, for the memory-read-line and memory-read-multiple commands, the 106 will target disconnect after a cache line (32-bytes) of data is transferred. This is because the 106 only has a 32-byte PCI-to-System-Memory-Read Buffer (PCMRB)."

That's the buffer we were talking about with Mr. Teter?

A. Correct, yes.

Q. And, "Once it is filled, it must be flushed before fetching the next cache line. It is the responsibility of the master to run another memory-read, memory-read-line, or memory-read-multiple transaction from where the data left off." Do you see that passage?

A. Yes.

Q. Okay. And that describes how the 106 worked, and it describes how the 105 worked, right?

A. Yes.

Q. Okay. So on both chipsets, once a line of

0115

data was transferred, the transaction would be ended and the target -- there would be a disconnect so that the buffer could be flushed, right?

A. Yeah. By -- yeah. By flushed, though, that -- I'm unclear what they mean in this context.

Q. Okay.

A. Nothing had to haen to be contents of that buffer before fetching the next cache line. Flush is probably not the right word. Invalidated is probably the correct word.

Q. Okay. All right. With that amendment, though --

A. Yes.

Q. -- what happened is --

A. Yep.

Q. -- a line of data would be --

A. Brought in.

Q. -- brought in, and then transferred, and then the transaction would be ended. And if the -- if the master wanted another line of data it had to initiate a new request --

A. On the PCI bus.

Q. -- and start a new transaction, right?

A. Yes.

Q. On the PCI bus, you said?

0116

A. Yes.

Q. Okay. And that's how both the MPC105 and the 106 operated, right?

A. Yes.

0119

Q. Okay. Now, the -- I guess the side box in which that table appears, MPC105 performance, that's -- that's describing the ability of the chipset to perform certain sorts of transactions more efficiently than what would otherwise be the case, right? Do you have trouble with that?

A. Yeah.

Q. Let me try a better question.

A. Okay.
Q. Okay. And let's focus on the one that Mr. Teter focused on, the PCI-to-memory --
A. Okay.

0120

Q. -- pipelined burst-read --
A. Uh-huh.
Q. That shows an initial transaction --
A. Yes.
Q. -- of -- which would have occupied -- nine plus seven, so 16 clock cycles, right? And then a second transaction of four plus seven or 11 clock cycles?
A. Okay.
Q. Right?
A. Yes.
Q. And by using the pre -- the speculative snoop feature the MPC105 was able to reduce the time it took to transfer -- to do two transactions, transferring two lines of data, from, I guess, 36 clock cycles to 31 clock cycles?
A. From 32 clock cycles to 27, is that -- right, two 16 cycle --
Q. I'm a philosopher, not an engineer. I trust your --
A. Yeah, okay.
Q. -- arithmetic way more than mine.
A. Basically from having two 16-cycle transactions down to a 16 cycle and an 11 cycle.
Q. Okay.

0121

A. Is that --
Q. Right.
A. Yes. Okay.
Q. And that was viewed as a performance benefit --
A. Yes.
Q. -- to the users of the system, right?
A. Yes.
Q. Now, if we could go back to the -- and I think you told Mr. Teter that those numbers you believe came from a simulator of some sort?
A. Yes.
Q. Okay. At this point you weren't actually in a position to test the chipset for this kind of thing?
MR. TETER: Object to form.

THE WITNESS: For the paper? I believe these came -- yeah. I mean, we might have been in a position to test that. I don't know. But I believe these numbers came from just doing a simulator.

Q. (By Mr. Brody) Okay.

A. These were like the best case numbers for these particular system configurations.

0130

Q. Okay. How many clock cycles does it take -- did it take to transfer the two lines of data on the MPC105 and 106, or at the least on these drawings, when the speculative read feature was enabled?

A. I count 27 cycles, PCI cycles.

Q. Twenty-seven cycles. And that consists of -- excuse me -- 15 for the first line --

A. Yes.

Q. -- to be transferred?

A. Yes.

Q. Okay. And then how many cycles are there between the end of the first transaction and the start of the second transaction?

A. One cycle.

Q. Okay. And then how long does the second

0131

transaction start -- take? I'm sorry.

A. I think there is 11 cycles. Eleven cycles, I believe.

Q. So what -- and the MPC105 would have been essentially the same?

A. Yes.

Q. All right. So what the speculative snoop and speculative read feature accomplished on the MPC105 was it shortened that latency between the two transactions by four clock cycles?

MR. TETER: Object to form.

THE WITNESS: Yes.

Q. (By Mr. Brody) And you folks consider that an enhancement to the performance of the system, and that's why you put the feature in the system, right?

A. Yes.

Q. Okay. And I assume if you could have eliminated all of the latency between the two lines that were being transferred, that would have been better still, right?

MR. TETER: Object to form.

THE WITNESS: Yes.

Q. (By Mr. Brody) Okay. It would be better -- how much latency is there in the transaction where the speculative read was enabled between the two lines?

0132

A. I'm sorry. Ask it again.

Q. Sure. I guess what I want to ask is between the time that the last data unit of the first line --

A. Yes.

Q. -- transfers, the transfer of that data unit is concluded, and the time when the first data unit of the next line --

A. Okay.

Q. -- begins --

A. Yes.

Q. -- how many clock cycles are there?

A. Four cycles.

Q. Four?

A. In this drawing there are four in between.

Q. Isn't there eight?

A. No. Those are processor clock cycles.

Q. Oh, okay. Oh, I see. I see what you're looking at.

A. All the -- the dash lines are processor clock cycles. But PCI clock cycles are twice as long.

Q. Okay.

A. And that's what those numbers are. Does that make sense?

Q. Yes, it does make sense. And how many clock cycles, PCI clock cycles, were there between the time

0133

when the last data unit of the first line -- when its transfer ended and when the first data unit of the second line, its transfer began when the snoop -- speculative read was not enabled? That should have been --

A. Eight.

Q. -- eight?

A. Yes, eight cycles.

Q. When the speculative read was enabled on the MPC105 and 106, I think you told us before that each data unit in a line would transfer on a successive clock cycle. So it would take one clock cycle for the first one to transfer, the second one would transfer on the second clock cycle, the third one on the third clock cycle, and so on, right?

A. Yes.

Q. Okay. And then in order for the next data unit, the first data unit of the second line to transfer --

A. Yes.

Q. -- that would be an additional five clock cycles, I guess. So it would be transferred on the 13th clock cycle?

A. Oh, I see what you are saying.

Q. There would be four cycles of latency and then

0134

one cycle for the unit to transfer?

A. Yes, on the 13th cycle, yes.

Q. Okay. And then the -- I guess it would be the second unit of the second line would transfer on the 14th clock cycle --

A. Fourteenth.

Q. -- and so on, through the last unit of that line of data.

A. Yes.

Q. And then if there were a third line of data, the first unit would transfer --

A. With --

Q. -- on the --

A. You could add five --

Q. Five to whatever --

A. -- to the number -- to the last number, right.

Q. Okay.

A. So there is four PCI cycles in between.

The feature is similarly described in the testimony of the other Motorola witnesses (Deposition of Michael Garcia, 36:11-37:16, 38:19-39:21, 42:13-43:12, 66:17-70:11; Deposition of Christopher Bryant Dep., 10:22-12:6, 15:6-16:2, 19:11-22, 33:13-23, 36:25-39:4, 45:18-60:17, 61:8-14, 62:1-64:6) and in multiple documents, including, as is discussed more fully in the referenced testimony, Defendant's Deposition Exhibits 61, 66, PDX 11, 13.

Elements of claims 1 (which is included in claims 7 and 8), 9, 21 and 26 read as follows:

1a. *sequentially transferring data units between said bus master and said secondary memory beginning at a starting memory location address in said secondary memory address space and continuing beyond an L-byte boundary of said secondary memory address space, said sequentially transferred data units including a last data unit before said L-byte boundary and a first data unit beyond said L-byte boundary; and*

9a. *sequentially transferring at least three data units between said bus master and said secondary memory beginning at a first starting memory location address in said secondary memory address space and continuing sequentially beyond an L-byte boundary of said secondary memory address space; and*

9c. *all of said transfers of data units in said step of sequentially transferring, occurring at a constant rate*

21a. *means for sequentially transferring data units between said bus master and said secondary memory beginning at a starting memory location address in said secondary memory address space and continuing beyond an L-byte boundary of said secondary memory address space, said sequentially transferred data units including a last data unit before said L-byte boundary and a first data unit beyond said L-byte boundary; and*

21b. *means for initiating a next-line inquiry, prior to completion of the transfer of the last data unit before said L-byte boundary, to determine whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being a line of said secondary memory which includes said first data unit beyond said L-byte boundary.*

26a. *means for sequentially transferring at least three data units between said bus master and said secondary memory beginning at a first starting memory location address in said secondary memory address space and continuing sequentially beyond an L-byte boundary of said secondary memory address space; and*

26b. *means for, prior to completion of the transfer of the first data unit beyond said L-byte boundary, determining whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being the line of said secondary memory which includes said first data unit beyond said L-byte boundary,*

26c. *said means for sequentially transferring, transferring all of said data units at a constant rate.*

The PCI specification (v. 2.0, DDX Exhibit 64, at OPTINVIDIA 001548) states that "[t]he basic bus transfer mechanism on the PCI is a burst. A burst is composed of an address phase and one or more data phases." (*See also* Section 3.2.1 at OPTINVIDIA 001549).

Taking into account both the obvious meaning of the claim elements 1a, 9a, 9c, 21a, 26a, and 26c above, and the quotation from the PCI Specification, I read all of the claim elements above to require that a single transfer span cache lines. Additionally, with respect to claim elements 21a, 21b, 26a, and 26b, as discussed more fully in my Infringement Report of May 16, 2006, I believe that the result achieved by the disclosed means are as follows:

21a/26a: The result achieved by sequentially transferring data in this way is that multiple lines of data can be transferred between memory and a PCI master at a constant rate, and the transfer can continue until the master elects to end it. '906 Patent, 6:8-23. This "mechanism can help PCI bus masters to achieve the full promise of high-speed data transfers afforded by the PCI bus burst transfer protocol." '906 Patent, 6:2-4.

21b/26b: The result achieved by initiating a next-line inquiry prior to completion of the transfer of the last data unit before the L-byte boundary of the line currently being transferred is that the transfer of the next-line is not delayed by the need to wait for the inquiry of that line to conclude.

The testimony of the Motorola witnesses and the documents discussed in that testimony clearly indicate that neither the Motorola MPC105 nor the MPC106 function in that manner or achieve these results - each disconnects at the end of each line, and must issue a new transaction to continue. Accordingly, those products cannot anticipate the asserted '906 Patent claims.

### 7.2. The MPC 105 Is Not Prior Art To The Claims At Issue

Information about the MPC105 bridge chip was presented at the Hot Chips VI symposium in August, 1994, (Reynolds, p. 9, DDX 66), and appeared in the proceedings of that conference. Information was also presented at the Intl. Conf. on Computer Design (ICCD) in October, 1994. (Reynolds, p. 13, DDX 61). Reynolds did not know if Motorola had a working MPC105 by October, 1994. (Reynolds, p. 19).

The following testimony by Reynolds describes his knowledge of the timing of the development of the MPC105 bridge chip and its status with regard to functionality of the MPC 105 chipset:

Q. No, that's fine. When did you start working on the MPC105 project?

A. For sure in May '93. I guess it would be questionable how much time before that was spent officially on it. We did some background investigation -- well, I can go -- from -- well, from June of '91 through, I don't know, October-ish of '91 I was working on just some Motorola-specific projects. And then they -- Motorola, Ale, IBM -- signed their AIM Alliance in, I think, October '91.

0012

So there was probably some investigation work before this date, but I think this is approximately May -- sorry -- May '93 is approximately the date when we officially started as a real project with actual coding, getting, you know, directory structures and databases set up, things like that.

Q. In May of '93 did you plan for the MPC105 to have a speculative read feature?

MR. BRODY: Object to the form.

THE WITNESS: I don't know at what point we decided to put that feature in, but I believe it was either early to middle of the project.

0013

Q. As of the date that Exhibit 61 was submitted to the IEEE, did Motorola already have a working MPC105 chip?

A. I don't know. I will clarify the previous statement that I believe this is an actual die photo, but I guess it's possible that they could have just taken the database that represents this and gotten graphical representation of that -- basically a plot, and taken a picture of it. But I believe that this -- in this picture I believe that this is the actual die.

Q. Do you recall how long it took you from the time that you had designed the MPC105 functional description to actually figure out how to implement it in these 256,000 devices on a single die?

A. No. I would -- if I understood the question, I would say that that was from May '93 we probably had a specification we were going to work towards, if not a, you know, final, absolute specification, but ideas. And then until we finished, which was -- I don't remember when -- a year-ish later. And so the -- repeat the question because that wasn't --

MR. TETER: Can you reread the question?

THE WITNESS: -- the question exactly, please.

(The requested portion was read back)

THE WITNESS: Okay. So what I was going to say was, that from the time -- this May '93 date to whenever we -- we finished and they, what we would call tape-out -- which means send the design database to be

0021

manufactured -- that to implement these 256,000 devices took most of that time. But they came on sequentially, I guess, you would say, where certain things were --

would have been implemented in a portion of these devices early on from -- you know, in 1993.

Some of the functions would have been late '93 when they actually get coded is what I meant by -- it's kind of like writing software -- from the time they actually get written down in the RTL design language and then they try -- from that point on -- there is more work before you -- before the date when we send the design database, which involves verifying that the equations you've written are implementing the functions you wanted them to.

There is also verifying that the equations you have written get -- are going to be produced in a manner that's fast enough to meet your design frequency target, which is basically how fast the chip is able to execute functions.

And so all of the -- all of the devices would have been implemented, in some fashion, months before we actually taped out. But it's likely that there were a few that -- a few specific -- like the exact devices that ended up in the final design were not the exact same ones from a few months before

0022

because there are -- we call them bugs. But there are problems in either the -- in the way the equations were written that is not actually doing the function specifically the way you needed it to, or you may not -- may not be implementing it fast enough to meet the speed that you wanted it to. So you would redesign certain small sections of the logic to fix the problems that are found along the way.

Q. (By Mr. Teter) You mentioned a coding process. When did you code the speculative read feature for the 105?

A. What?

Q. When.

A. When?

Q. When did you code it, yeah.

A. I don't remember exactly in the project when it was.

Q. Who were the verification engineers that you acknowledge in Exhibit 61?

A. Adrian Harris, Glen Wilson, and Michael Carlson.

0026

Q. At a high level, what was their role in the MPC105?

A. To develop -- based on the specifications of what we told them the functionality was they were to write tests that tried to exercise the functions in the chip -- basically the software that we wrote -- to verify that it was performing as specified.

Q. In Exhibit 61 you acknowledge the verification engineers that worked on the MPC105 project. Was the work of the verification engineers completed as of the time you submitted Exhibit 61 to the IEEE?

A. Yes, but we continue even after we tape-out to run random verification tests just to possibly uncover things that might not have gotten tested during the project.

Q. Why did you include the ability to turn off the speculative read feature?

A. Two reasons. Primarily it was so that -- we

...

And so we envisioned some systems that might not really want this feature because it would

0037

actually degrade their performance. And the second reason was, in the design of the parts -- or in the testing of the parts in the design verification, this section of the chip maybe had a larger percentage of bugs than other features we had put in. And so having the ability to disable it would not prevent systems from coming up and actually working. All it would be is a performance loss on such a system. But it could still become a real system that's functional and could get sold in the marketplace.

Q. Okay. So during this period of time from, I guess, July of '93 through roughly October of '94, work on the project had gotten through the steps that are listed on Exhibit 60; is that correct? The architecture had been completed, you implemented the -- you completed the logic implementation, verified the timing of the central control unit --

A. Right.

Q. -- and debugged the transfer control unit and configuration --

A. Right.
Q. -- register?
A. Yes.
Q. Okay. I would like to understand a little bit
0097

Q. Okay. Now, how long -- do you have a recollection as to how long the logic design took for this project?
A. Not an exact recollection, no.
Q. Can you give me an order of magnitude?
A. Yes. Between -- well, at least six months.
Q. Okay.
A. Probably up to, I guess, 16 months, I would
0099
say. That's the best order is between six and sixteen months.
Q. Okay.
A. Yeah.
Q. And then how does -- how long does the -- did the verification physical design process take, to the best of your recollection?
A. Well, okay, so for verification, similar amounts in that once some of -- once some of the logic is ready, that gets tested and has more of --
Q. Kind of rolls with the design?
A. It rolls with it, so, yes.
Q. Okay.
A. Sorry.
Q. I'm sorry.
A. Let me clarify the 16 months. That would be -- I mean, the logic design goes the whole time, but at the end of the project the amount of logic design is very minimal. It's only just bug fixes and timing fixes.
Q. Uh-huh.
A. And so that 16 months was kind of from start to end that you could say we did logic design for 16 months. But the last three months -- and maybe this is where you were leading. The last three months is very
0100
heavily the physical design intensive. But the verification is intensive the whole time once some logic is on line. So a year probably. Does that help your -- does that answer your question?
Q. What I'm really trying to get to is try to

figure out what the time horizon would have been before you got to the tape-out.

A. Okay. It's not exactly like this, but you could think of it as, you know, nine months of logic design, six months of -- well, four to six months of verification, and three months of physical design. But the physical design overlaed with verification.

But that first number I gave you for the logic design, you have to understand that there is preliminary work done on verification and physical. I mean, otherwise you will never make those numbers. You won't do physical in three months. But most of the work is all just -- it's really like prototyping because you know the logic is not the final logic. But you are getting the flow worked out so that when the final logic is available you can quickly get to the physical design.

Q. Okay. So, if I'm understanding the numbers you are giving me --

A. Okay.

0101

Q. -- it sounds like you folks were roughly ready to tape-out the chip so that you could send it off to manufacturing somewhere on the order of a year plus after you started working on the project?

A. Yeah. I would say so.

Q. So that would put it somewhere in middle of 1994?

A. That's what I believe.

Q. Okay. You said that during the period, at least, I suppose, August to October you were doing debugging of these boards; is that right?

A. Uh-huh, yes.

Q. Okay. And I think you said that part of the reason why you had an ability to turn the pre-snoop feature, the pre -- the speculative snoop feature on and off was because that particular feature was very buggy?

A. Yes.

Q. Okay. And you wanted to give your customers the ability to disable the feature so the -- the boards would work, right?

A. Yes.

0107

Q. Okay.

A. That was one of the reasons.
Q. Yeah. You finally disabled the -- that on/off feature in the MPC106 product, is that right, or at least for the --
A. For the memory read and multiple command I believe that it didn't -- it no longer referenced the configuration, but --
Q. Okay. Do you know why you disabled the configuration bit for the memory read multiple on the MPC106?
A. I believe it was because we felt like more PCI master parts were using memory read multiple now than the way it was intended in the original spec. And, secondly, we now had more verification and felt like we didn't have bugs or potential bugs with that feature anymore.
Q. Uh-huh. So you had a reliable speculative snoop feature by the time the MPC106 was released?
A. Yes.
Q. Okay. But I gather it was sufficiently unreliable when the 105 was released that you felt, at least in part, for that reason, that you needed this on/off capacity?
A. In part.
0108
MR. TETER: Object to form.
THE WITNESS: In part.
Q. (By Mr. Brody) Now, the MPC106 that I have been referring to, that's the successor product to the MPC105, right?
A. Yes.
Q. Okay. And I gather, since you first started talking to Mr. Teter in connection with this deposition, you have been giving a little thought to the relationship between those two products?
A. Yes.
Q. Okay. And am I correct that to the best of your recollection, aside from the difference with respect to the memory read multiple command, the ability to turn that on and off --
A. In the 105?
Q. In the 105, but not the 106.
A. Yes.
Q. The 106 executed the speculative snoop and speculative read features essentially the same way as the 105?

A. The best I recall. I don't recall any other differences.

Q. Mr. Brody also asked you whether or not the MPC105 speculative snoop, speculative read feature -- strike that.

0138

Mr. Brody also asked you questions about whether or not the MPC105 -- strike that. Let me try and get this right.

Mr. Brody asked you some questions about whether or not the MPC105's speculative read feature was, quote, very buggy. Do you remember that --

A. Yes.

Q. -- question and answer?

And I just want to make sure we're -- that I'm clear on this. The boards that you shipped with the MPC105, did the speculative read feature work in the ways that you have testified here to today? Did it function?

A. Yes.

Q. And was it perfect? Did it function perfectly in every possible instance?

A. No. No.

Q. In the basic scenario of a PCI master reading data from secondary memory, did the speculative read feature work as you have described here today?

A. Yes.

Q. What were the circumstances where it was buggy? Do you remember any of the bugs at all?

A. I don't remember any specifics, but it -- it typically -- my recollection is that it involved the

0139

similar -- the same addresses being in other buffers also within our chip, and so they were more -- they were more intricate than just the PCI master being there and doing a read. Other things had to be happening in the system in parallel, at the same time, whatever, that caused -- the logic just hadn't accounted for that particular case. But I don't remember a specific instance.

Q. But the basic functioning of the MPC105 where there is one PCI master reading from secondary memory, that actually worked in the MPC105; is that right?

A. Yes.

Q. It was able to perform the speculative read function just as we have described here today; is that right?

A. Yes.

Q. When you were working on the MPC105, did you understand that the reason there would be the disconnect at a cache line was because you only had a buffer that was one cache line long?

A. Yes. You might say that it was also because -- yeah. We didn't -- and also because we wouldn't partially fill it with the next line was the other reason we had to target disconnect.

Q. If you understood that the target disconnect was because you had a buffer that was one cache line long, why, then, didn't you just have two PCMRB buffers?

A. I think it was because the additional area and complexity of controlling the second buffer was more than we wanted to do. We also believed that the

0141

performance benefit of the speculative reads was sufficient for the trade-off of -- of the area for a second one. It starts to -- I guess even with one it looks like a cache. The more you have, the more states -- or more you have to -- you have to track for each individual one that stayed in the system. I was going to say something else.

Q. Mr. Brody asked you some questions about whether eliminating the latency between data transfers in the current line and the next line, whether that would have made things better still. Do you recall those questions?

A. Yes.

Q. And you indicated that it would be better still to remove that latency. Do you recall that?

A. Yes.

Q. At the time you were working on the MPC105, did you have an idea how you could remove that latency? Maybe it's a bad question. I'll withdraw it.

At the time you were working on the MPC105, did you have any understanding about whether you could remove that latency, albeit at some additional cost?

A. Yes.

Q. And what was that understanding?

0142

A. That that latency could be removed. Just that if we didn't -- it's kind of -- it's complicated. So we would -- we would -- it's dependent on -- yeah, we thought about how we could do it, but determined that it was not -- I don't know -- not feasible. Well, we -- I don't how to answer this exactly, correctly.

I mean, it's -- there is a number of components. There is the speed of the memory, there is whether you're hitting an open page because the PCI spec said that you -- between data beats transfers you had to be eight PCI cycles or less. And so we didn't believe we could meet that in all cases. We didn't want to detect that we could do that in all cases. And so we would just always disconnect at the end of the cache line under the assumption that the master, if it was doing more than one cache line, could get back on the bus reasonably quickly.

And so to answer your question did we know how to eliminate, probably. But it wasn't -- it wasn't for our system what we thought. It wasn't really in our best interest to do that. So we didn't pursue it with a bunch of diligence to actually make it work, is what I would say, to be able to -- as an engineering trade-off we said this is sufficient.

So I -- I wish I could say yes or no.

0143

Yes, we had to come up with how to do it or, no, we didn't even think about it. But I can't really say either way. I don't -- I don't specifically recall us sitting down one day and saying, yes, hey, we could eliminate more of these cycles by doing such and such, but --

Q. When you say it was an engineering trade-off, do you mean between performance and what it would cost --

A. Yes.

Q. -- to get there?

A. Yes.

Q. And I think you looked at this before in Exhibit 61 but you had said that the MPC105 enabled -- allowed streaming of large amounts of data with very minimal delay on the PCI bus enabling the system to utilize the full bandwidth of the PCI bus. Was that your understanding of what the speculative read feature allowed?

A. Go back to which exhibit?
Q. This was 61. This is the paper that you wrote in '94.
A. Sorry. When it said what?
Q. This is right above, "Other features," above the die, above the picture of the die.

0144

A. Yes, enabling the -- yes. The answer to your question is yes.
Q. So was it -- was it your view at the time that the speculative read feature allowed transactions to be fast enough?
A. Yes. And at the time we also were not convinced for the 105 that there were devices that were using memory read multiple as a command. There probably were, but it was not -- I don't think it was prevalent. And so that also factored into our decision to --

0148

Q. I think you told Mr. Teter that you thought there might have been ways to do that, but essentially you elected not to figure out which ways would actually work and reduce them to practice because you thought that there were other engineering issues that were -- or you thought the cost benefit wasn't there; is that correct?
MR. TETER: Object to form.
THE WITNESS: Yes, that's a -- yes.

0149

Q. (By Mr. Brody) Okay. I mean, did you --
A. Correct.
Q. I'm sorry?
A. Go ahead.
Q. You said -- at one point when you were talking about that you said that there were certain requirements from the PCI specification, including the need to have no more than eight cycles between the start of --
A. That's between any two data transfers.
Q. Okay.
A. So -- I'll just continue with the clarification. From FRAME to the first data beat there is no restriction. But from data beat to subsequent data beat or data transfer there was an eight PCI clock maximum.

Q. Okay. And I think you said you weren't sure that you could meet that in all cases, so you just had a disconnect at the end of each cache line in order to assure --

A. Yes.

Q. -- that you would; is that right?

A. Yes.

Q. Okay. You said you thought that customers had operating boards at some point prior to October of

0150

1994 -- or, I guess, October 21st of 1994.

A. I believe so, yes.

Q. Do you know how far in advance?

A. No.

Q. You said that the boards that were shipped were capable of performing the speculative read and speculative snoop functions as described. Did I hear that correctly?

A. Yes.

Q. But you were also concerned that they couldn't do that reliably, right, and that's why you allowed your customers to turn off the feature?

A. That was part of it, yes.

Q. Okay. Do you have any idea how often or how frequently boards were incapable of performing those functions?

MR. TETER: Object to form.

THE WITNESS: No, I do not know.

MR. BRODY: That's all I have.

MR. TETER: One last question.

FURTHER EXAMINATION

0152

FURTHER EXAMINATION

BY MR. BRODY:

Q. Did you develop -- did you make a version of the MPC105 that eliminated the latency between lines in a multiple line transfer?

A. In other words, made it zero --

Q. Right.

A. -- between -- no.

Q. Okay. Did -- were any drawings prepared that would have -- any logic designed that would have

0153

accomplished that?

A. I don't recall any, no.

Based on the above testimony by Reynolds (and the remaining testimony appearing in his deposition transcript), I do not find any basis to conclude that whatever pre-snoop invention appears in the Motorola MPC105 can be dated to any specific date whatsoever. There is no indication that the pre-snoop feature of the MPC105 was ever debugged or working reliably, nor is there any evidence that any bugs were not design errors of the type discussed above in Section 3.5. In fact, it seems quite clear that the "corner cases" discussed above were not properly accounted for in the MPC105, and Reynolds was explicit in testifying that an enable bit was added to the chipset precisely because the feature was so buggy that there was concern that boards incorporating the chipset would not work if the feature was enabled. A pre-snoop feature was apparently working in the MPC106 product, whenever that was debugged, although it was not the pre-snoop feature of the '906 Patent. No date has been presented to me for the fully functional operation of the MPC106 product, but based on the evidence I have seen, that would be the earliest date that would clearly establish conception of Motorola's pre-snoop feature.

Garcia Testimony

[redacted]

[redacted]

Bryant Testimony

Christopher Bryant also testified regarding the conception and reduction to practice of the MPC105. The most pertinent sections of his testimony are excerpted below. Nothing in his testimony changes my opinion with regard to whether the MPC105 constitutes a prior invention of the '906 invention. Rather, his testimony indicates that at some point after the Spring of 1994, when he left the project, Motorola added the bit allowing the pre-snoop feature to be disabled and he also confirms that the feature may have been disabled in the demonstration chipset shipped in the Fall of 1994.

[redacted]

Q. Okay. And I think you told Mr. Cunningham that you were not in the lab and weren't working with the chipset when you were verifying that the silicon actually operated the way you expected it to?

A. That's correct.

Q. So if there were changes to the way that enabling bit functioned after the silicon came back, would you have been aware of those?

A. Yes.

Q. Those would be recorded in the User's Manual, right?

A. Yes.

0047

Q. Okay. Could you look at page -- you have got the User's Manual there. It is Defendant's Exhibit 65. Could you look at Page 12443, please, NV12443, which is Page 7 --- I am sorry, 1244, which is Page 7-6 of the User's Manual.

A. Okay.

Q. Okay. You got that. And you see, that's a list of PCI bus commands?

A. Uh-huh.

Q. For the MPC105, right?

A. Yes.

Q. And you see the memory read multiple command there?

A. Yes.

Q. If you look at the definition of the command, it says: "The memory read multiple command functions the same as the memory read command on the MPC105." And then it says "If prefetching is desired, speculative read should be enabled. See Section 8.1.3.2.1." Do you see that?

A. Yes.

Q. Does that refresh your recollection that you had to set the bit to enable in order to do a prefetch even on a memory read multiple?

A. No. I mean it says it, but I don't remember

0048

that.

Q. So when people -- and presumably, the chips that operated in conformity with the description in the manual.

A. I am sorry?

Q. Did the chips that operate in conformity with the description in the manual?

A. Of the MPC manual or the PCI manual?

Q. The MPC manual.

A. So far as I know.

Q. So far as you know, at the time the manual was written, this is an accurate description for this to say, if you wanted to pre-snoop even on the memory read multiple command, you would have to set that bit to enable, right?

A. As far as I know, right.

Q. Do you recall learning from anybody in the team who had been working in the lab to verify the silicon that the pre-snoop feature was unreliable?

A. No.

Q. Okay. Do you recall learning that it was important to set the bit to permit it to be disabled because otherwise your customers couldn't be assured that the chipset would actually work?

A. No.

0049

Q. Do you know why the bit was set so that you could disable the pre-snoop feature even on a memory read multiple command?

A. No.

Q. I am going to read you a passage from Mr. Reynolds' deposition given on March 30th. And what I would like to know is whether this refreshes your recollection in any way as to why that bit was there to let you disable pre-snoop even on a memory read multiple, okay?

A. Okay.

Q. He says, this is sort of in the middle of an answer. "One reason was, in design of the parts or in the testing of the parts." So that would be in this period of time when you weren't directly involved in the project, right, that testing period?

A. I don't know. I mean it could be he is talking about verification testing.

Q. Okay. Well, "in the testing of the parts, in the design verification, this section on the chip maybe had a larger percentage of bugs than other features we had put in. So having the ability to disable it would not prevent systems from coming up and actually working. All it would be is performance loss on such a system. But it could still be a real system that is functional

0050

and could get sold in the marketplace." Does that your refresh your recollection as to the reason why the pre-snoop could be disabled even on a memory read multiple?

A. No.

0062

Q. Do you ever remember hearing from the team members or any customers that the pre-snoop feature was unreliable?

A. I didn't deal with any customers so, no. And from team members, I don't remember.

Q. After the MPC105 taped-out, do you remember

0063

hearing any feedback with regard to the pre-snoop feature?

A. No.

## 8. US Patent 5,581,714 to Amini et al. Does Not Invalidate The Claims At Issue

It is alleged in the Colwell report that U.S. Patent No. 5,581,714 to Amini et al. (the '714 Patent) either anticipates the '906 Patent or renders it obvious. For reasons explained in this section, I do not agree.

### 8.1. The Amini Disclosure

The '714 Patent describes bus interface logic ("PCI host bridge 20") (which I refer to as the "host bridge" or "bridge")) that interfaces between an external bus (the PCI bus) and a CPU bus (the "S-bus" or "system bus"). The only burst transfers that the system bus permits are burst transfers of 16 bytes (4 double words), aligned on a 16-byte boundary. ('714 Patent, 3:10-12). (It appears that in some cases, read bursts from the S-bus to the PCI bus may also be 8 bytes and on 8-byte boundaries ('714 Patent, 20:59-21:50 and table 2.)) The PCI bus permits a burst to be of any length and start on any byte boundary. ('714 Patent, 3:3-7).

The processor, cache and memory complex (which I refer to as the "host system") are attached to the system bus. In the preferred embodiment, the processor is an Intel i486. There is very little description of the processor, cache and memory complex, and no discussion of cache characteristics or cache snooping.

.
The bridge functions as follows:

> When a master connected to a PCI bus needs to transfer data to a component or device connected to a system or CPU local bus, a two-step procedure must be used. (System buses, for example S-bus 16, and CPU local buses both conform to X86-type bus architecture, and thus it will be hereinafter referred to as CPU local bus architecture.) During the first step, the host bridge that connects the PCI bus to the CPU local bus is a slave for a data transfer on the PCI bus. For the second step, the host bridge becomes a master for a read or write cycle, whatever the case may be, on the CPU local bus and the device or component to which the data transfer is targeted is a slave for this particular data transaction. ('714 Patent, 9:36-48).

The heart of the PCI Host bridge is a FIFO buffer:

FIFO buffer logic 102 includes a first-in-first-out buffer (FIFO) (not shown) which is capable of storing a 16 byte (4 DWORD) data string. The FIFO is capable of read-prefetching data from S-bus 16 and write-posting data into it from PCI Bus 22. The FIFO read-prefetches data by anticipating that a read burst transfer will read data from consecutive addresses, and thus, "prefetches" the data from those addresses before the particular master initiating the transfer requests to read the data. Write-posting occurs when the FIFO accepts data from consecutive addresses during a write burst transfer from a PCI master, and thus, "posts" the data from the master before the host bridge transfers it to the slave connected to S-bus 16. ('714 Patent, 10:61-11:6).

The FIFO has only a single 16-byte buffer:

During either a posted-write or read-prefetch burst transfer initiated from PCI bus 22, when the data being posted or fetched, respectively, fills up the FIFO, the burst transfer must be terminated or valid data will be written over in the FIFO. ('714 Patent, 11:31-35).

The PCI host bridge has two interfaces - to the PCI bus and the S-bus. When a device on the PCI bus wants to read from the main memory ("base system memory", 32):

In operation, a PCI master initiates a read burst transfer during a PCI command/address phase by asserting a Memory Read Long or Memory Read Multiple command on the C/BE line of the PCI bus 22. During the same command/address phase, the master transmits a memory address on the AD line of PCI bus 22 that indicates the read transfer is directed to a slave connected to S-bus 16. ('714 Patent, 18:1-7).

When the data arrives in the FIFO, the PCI bus master starts to read it:

During a read burst transfer initiated by a master connected to PCI bus 22 and targeted to a slave attached to S-bus 16, host bridge 20 read-prefetches data from the slave into the FIFO. TRDY is asserted when the data that the PCI master is attempting to read from a particular address is present in the DWORD of the FIFO that corresponds to bit numbers 2 and 3 of the address. Pacing logic 110 also provides this function of asserting TRDY for read burst transfers initiated from the PCI bus 22.

The FIFO ready logic block 124 functions the same way for read transfers as it does for write transfers. Thus, when a read transfer is initiated by the PCI master for the S-bus 16 and the FIFO is empty, S_DATA becomes high which causes FIFO_AVLBL to be high throughout the data transfer. ('714 Patent, 15:59-16:6).

What the "pacing logic" does is, rather than disconnect the PCI master and terminate the transaction when the master is attempting to write to a full FIFO buffer or read from an empty one, it just de-asserts TRDY, until the buffer is again empty or full as needed. ('714 Patent, 11:53-12:2)

In summary, on a PCI memory read multiple (MRM), the read prefetch logic loads the FIFO buffer. When the FIFO buffer double word is filled, the PCI bus (master) is allowed to read from it. Whenever the target word that the PCI bus is trying to read is not present, TRDY is deasserted by the bridge. When the FIFO becomes empty, the bridge reads another line (16 bytes) of data from the host system. Thus the PCI master can issue a single MRM, and the bridge will transfer line after line from the memory to the PCI master without terminating the transaction. (*See also* '714 Patent, 16:37-45, 54-60).

Note that the FIFO consists in the '714 Patent of only a single 16-byte buffer. ('714 Patent, 10:61-63). Notwithstanding reference to a "double word interleaved structure" or "ping-pong structure", ('714 Patent, 11:6-8), the patent specification consistently describes the FIFO as a single 16-byte buffer. (*See* '714 Patent, 11:25-36). Thus, an MRM proceeds by filling the FIFO from the S-bus side, reading out the contents on the PCI side, and repeating this sequence until the PCI bus master terminates the transaction. Note specifically that because there is only a single buffer, only one line of data can be in transit at any given time. So any snoop that occurs in the host system in the course of filling the FIFO from the S-bus is not a pre-snoop, because it will take place after the preceding line has been completely transferred to the PCI master.

Amini, in his deposition, appears at several points to recall that the FIFO had more than a single 16-byte line of storage, (*see, e.g.,* Amini Dep. at 36, 55, 57, 58), but ultimately testifies, the patent only appears to disclose a single buffer and that he has no clear independent memory to the contrary:

46:11-47:15

Q Mr. Amini, first I just want to understand a little bit about the nature of your testimony today.

Mr. Teter walked through the patent and a lot of the features of the bus with you and you did your best, I think, to understand what was in the patent.

The question I wanted to ask was: Were you --

Do you have independent recollections of how the bus worked or were you just reading the patent and doing your best to understand it?

MR. TETER: Object to form.

BY MR. BRODY:

Q Or some combination of the two?

A I guess I was doing -- I don't know if I can distinguish, but -- conceptually, I recall some of the concepts, but I can't say after this many years that I remember, you know, many of the details.

So for that, I had to really go through as we were walking through with Mr. Teter.

Q So to understand the implementation of the bus, you have to look at the patent and read it just like anybody else would read it?

MR. TETER: Object to form.

THE WITNESS: If I were to start today, I would probably have to go back and, you know, understand the specifications and the issues to come up with an implementation.

57:11-58:2

Q Okay.

A This is a long time ago. If I remember correctly, I would remember multiple buffers in this design, multiple 16-byte buffers in this FIFO. But I can't see it here.

Q Do any of the diagrams show multiple 16-byte FIFO buffers?

A Not that I can decipher from -- this is, obviously, some logic design that takes a little bit of looking at to decipher and understand. I can't see anything quickly that refreshes anything in a very specific way.

But logically, it would be technically very difficult to write or read at the same time with only one buffer, as referenced -- as described in someplace else that we went through as ping-pong. That's why I'm struggling with that.

63:2-5

I only have what's in here to go by in terms of refreshing my memory, but I just don't recall, you know, exactly what that might have or might not have included in the design. Just too long ago.

So basically, the patent says what it says, and Amini's recollection, even if it were to be considered to vary the disclosure of the patent, is not very good.

## 8.2. The Amini Patent Does Not Anticipate The Claims At Issue

The following claim elements are not present in the Amini '714 Patent, as explained below:

1b. *initiating a next-line inquiry, prior to completion of the transfer of the last data unit before said L-byte boundary, to determine whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being a line of said secondary memory which includes said first data unit beyond said L-byte boundary.*

8. *A method according to claim 1, wherein said next-line inquiry takes place concurrently with at least one of the data unit transfers in said step of sequentially transferring.*

9c. *all of said transfers of data units in said step of sequentially transferring, occurring at a constant rate*

21a. *means for initiating a next-line inquiry, prior to completion of the transfer of the last data unit before said L-byte boundary, to determine whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being a line of said secondary memory which includes said first data unit beyond said L-byte boundary.*

21b. *means for initiating a next-line inquiry, prior to completion of the transfer of the last data unit before said L-byte boundary, to determine whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being a line of said secondary memory which includes said first data unit beyond said L-byte boundary.*

26a. *means for, prior to completion of the transfer of the first data unit beyond said L-byte boundary, determining whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being the line of said secondary memory which includes said first data unit beyond said L-byte boundary,*

26b. *means for, prior to completion of the transfer of the first data unit beyond said L-byte boundary, determining whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being the line of said secondary memory which includes said first data unit beyond said L-byte boundary,*

26c. *said means for sequentially transferring, transferring all of said data units at a constant rate.*

Claim elements 1a, 8, 21a, and 26a call for a snoop to occur before the preceding line has been completely transferred. As explained above, that does not occur in the '714 Patent.

Claim elements 26c and 9c above also are not found in the Amini patent because the transfer occurs in spurts. A line is read from the host system into the FIFO, and then is transferred across the PC bus. Then wait states are inserted and then that sequence repeats. So from any single viewpoint in the system (e.g. the PCI bus master), the transfer consists of a sequence of double words going by, separated by idle periods.

Finally, with respect to claim elements 21a, 21b, 26a, and 26b, as discussed more fully in my Infringement Report of May 16, 2006, I believe that the result achieved by the disclosed means are as follows:

| | |
|---|---|
| 21a/26a: | The result achieved by sequentially transferring data in this way is that multiple lines of data can be transferred between memory and a PCI master at a constant rate, and the transfer can continue until the master elects to end it. '906 Patent, 6:8-23. This "mechanism can help PCI bus masters to achieve the full promise of high-speed data transfers afforded by the PCI bus burst transfer protocol." '906 Patent, 6:2-4. |
| 21b/26b: | The result achieved by initiating a next-line inquiry prior to completion of the transfer of the last data unit before the L-byte boundary of the line currently being transferred is that the transfer of the next-line is not delayed by the need to wait for the inquiry of that line to conclude. |

Because the Amini chipset inserts wait states following the transfer of each line, it does not achieve the results of the invention set forth in claims 21 and 26.

### 8.3. Lack of Obviousness based on '714 Amini Patent.

The Colwell report opines that the '906 Patent is obvious when considered in relation to Amini, but no obviousness argument is presented. Obviousness normally requires two or more prior art references and a motivation to combine. Without some indication of what Colwell has in mind here, I cannot respond. At this time, I am aware of no satisfactory argument that would justify a statement that the '906 Patent is obvious in light of the '714 Patent. If and when such an argument is presented, I may be asked to testify in reply.

## 9. IBM Technical Disclosure Bulletin alleged as Prior Art

It is alleged in the Colwell report that the IBM Technical Disclosure Bulletin (TDB) article (Vol. 36, No. 10, October, 1993, . 187-191) renders obvious the asserted claims of the '906 Patent. In this section, I discuss why I disagree.

The TDB article describes a pre-snooping design to work with the IBM Micro Channel Bus.

The TDB article fails to read on the asserted claims of the '906 patent.

I explain as follows, for specific claim elements.:

1a *sequentially transferring data units between said bus master and said secondary memory beginning at a starting memory location address in said secondary memory address space and continuing beyond an L-byte boundary of said secondary memory address space, said sequentially transferred data units including a last data unit before said L-byte boundary and a first data unit beyond said L-byte boundary;*

This element requires a transfer that spans cache line boundaries. The TDB article does not disclose such spanning transfers. The timing diagram on page 191 only shows four consecutive data transfers. An article about the Micro Channel (James O. Nicholson and Fred E. Strietelmeier, "New Micro Channel Features", Proc. Spring Compcon, February 26-March 2, 1990, 179-182) says that, in "a streaming data cycle . . . a *block* of sequentially ordered data is transferred during the CMD period. The address specifies only the starting address of the block." (emphasis added). Thus the Micro Channel architecture seems to support only block length transfers. The words "block" and "line" are typically considered synonyms in the context of cache memory. [Smith82]. Thus, after a block (or line), the transfer stops, and a new block-length transfer is requested. The "cache line address detect" in figure 1, therefore, detects that the new transfer is to a new block.

It would not be obvious to have transfers span cache line boundaries, since others of at least ordinary skill at the time (*e.g.*, those who designed the MPC105) did not construct a design for which transfers spanned block boundaries.

A second reason why this claim element is not present in the TDB disclosure, is that the article describes cases for which the transfer is between the bus master and cache, not between the bus master and secondary memory. ("On read operations from a store-in cache and if data is updated, the data is fetched from cache and put into I/O first-in-first-out (FIFO) IO buffers for output to I/O devices, but is left as valid and the data is not updated in cache." (p. 189)). "If any snoop address shows a line was updated, then the data is fetched from the store-in cache and not from memory." (p. 190).

The transfer from cache and not main memory is represented in the TDB article as a performance improvement, and it would not be obvious to do the transfer from main memory instead; the article teaches away from that solution.

1b. initiating a next-line inquiry, prior to completion of the transfer of the last data unit before said L-byte boundary, to determine whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being a line of said secondary memory which includes said first data unit beyond said L-byte boundary.

The TDB article is not specific about the timing of the pre-snoop, and does not necessarily initiate "a next-line inquiry, prior to completion of the transfer of the last data unit before the L-byte boundary." It would not have been obvious to time the pre-snoop as contemplated by this claim of the '906 Patent, since others of at least equal skill in the art at IBM (e.g., Amini) did not do so.

> 7. A method according to claim 1, wherein said bus master is a PCI bus master, wherein said first cache memory includes an instruction cache and a data cache, and wherein said host processing unit and said first cache memory are fabricated on a single CPU chip.

The TDB article describes a Micro Channel bus, not a PCI bus. It shows a unified cache, not separate instruction and data caches. It does not indicate whether the host processing unit and first cache memory are on the same chip.

> 8. A method according to claim 1, wherein said next-line inquiry takes place concurrently with at least one of the data unit transfers in said step of sequentially transferring.

The TDB article is not specific about the timing of the pre-snoop and does not specify that "wherein said next-line inquiry takes place concurrently with at least one of the data unit transfers in said step of sequentially transferring." It would not have been obvious to time the pre-snoop as contemplated by this claim of the '906 Patent, since others of at least equal skill in the art at IBM (e.g., Amini) did not do so.

> 9a. *sequentially transferring at least three data units between said bus master and said secondary memory beginning at a first starting memory location address in said secondary memory address space and continuing sequentially beyond an L-byte boundary of said secondary memory address space; and*

This element is not found in the TDB article for the same reasons as for element 1a.

> 9b. *all of said transfers of data units in said step of sequentially transferring, occurring at a constant rate*

This element requires a transfer that spans block boundaries without pause. The TDB article does not appear to disclose such spanning transfers. The timing diagram on page 191 only shows four consecutive data transfers. An article about the Micro Channel (James O. Nicholson and Fred E. Strietelmeier, "New Micro Channel Features", Proc. Spring Compcon, February 26-March 2, 1990, . 179-182) says: "A streaming data cycle is similar, but a *block* of sequentially ordered data is transferred during the CMD period. The address specifies only the starting address of the block." (emphasis added) Thus the Micro Channel architecture supports only block length transfers. After a block, the transfer stops, and a new block-length transfer is requested. The "cache line address detect" in figure 1 thus detects that the new transfer is to a new block.

It would not be obvious to have transfers span block boundaries, since others of at least ordinary skill at the time (*e.g.*, those who designed the MPC105) did not construct a design that spanned block boundaries.

> 21a means for sequentially transferring data units between said bus master and said secondary memory beginning at a starting memory location address in said secondary memory address space and continuing beyond an L-byte boundary of said secondary memory address space, said sequentially transferred data units including a last data unit before said L-byte boundary and a first data unit beyond said L-byte boundary; and

This element is not found in the TDB article for the same reasons as for element 1a.

> 21b means for initiating a next-line inquiry, prior to completion of the transfer of the last data unit before said L-byte boundary, to determine whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being a line of said secondary memory which includes said first data unit beyond said L-byte boundary.

This element is not found in the TDB article for the same reasons as for element 1b.

> 26a. means for sequentially transferring at least three data units between said bus master and said secondary memory beginning at a first starting memory location address in said secondary memory address space and continuing sequentially beyond an L-byte boundary of said secondary memory address space; and

See discussion, claim 1a above.

> 26b. means for, prior to completion of the transfer of the first data unit beyond said L-byte boundary, determining whether an N+1'th L-byte line of said secondary memory is cached in a modified state in said first cache memory, said N+1'th L-byte line being the line of said secondary memory which includes said first data unit beyond said L-byte boundary,

See discussion, claim 1b above.

> 26c said means for sequentially transferring, transferring all of said data units at a constant rate.

This element requires a transfer that spans block boundaries without pause. The TDB article does not appear to disclose such spanning transfers. The timing diagram on page 191 only shows four consecutive data transfers. An article about the Micro Channel (James O. Nicholson and Fred E. Strietelmeier, "New Micro Channel Features", Proc. Spring Compcon, February 26-March 2, 1990, 179-182) says: "A streaming data cycle is similar, but a *block* of sequentially ordered data is transferred during the CMD period. The address specifies only the starting address of the block." (emphasis added) Thus, the Micro Channel architecture supports only block length transfers. After a block, the transfer stops, and a new block-length transfer is requested. The "cache line address detect" in figure 1 thus detects that the new transfer is to a new block.

It would not be obvious to have transfers span block boundaries, since others of at least ordinary skill at the time (*e.g.*, those who designed the MPC105) did not construct a design that spanned block boundaries.

## 11. The Compaq TriFlex Chipset Is Not Prior Art To The '906 Patent.

I agree with Dr. Colwell's conclusion that Compaq's TriFlex chipset as *ultimately implemented* practiced the inventions covered by claims 1, 7, 8, 9 21, and 26 of the '906 Patent. I disagree with Dr. Colwell's conclusion that Compaq had conceived how to do so before OPTi did.

As I have noted previously, OPTi's engineers had a definite and permanent idea of the complete and operative pre-snoop invention as it is was ultimately applied in practice by the Spring of 1994. In contrast, there is no documentation disclosing a similarly complete conception by Compaq of how pre-snooping could be used to transfer multiple cache lines of data until, at the earliest, the October 14, 1994 application that ultimately led to United States Patent No. 5,634,073. (Defendant's Deposition Exhibit 182.) By the time that application was filed with the United States Patent and Trademark Office, OPTi had already received functioning silicon for its pre-snoop feature.

Preliminarily, I understand that, as a matter of law, neither Dr. Colwell nor I may rely on the uncorroborated testimony of the inventors of Compaq's pre-snoop feature (that is, Messrs. Thome and Collins) as to when they conceived how that feature would operate in practice or what the nature of their conception was. To my understanding, no testimony has been provided as to the conception and reduction to practice of the inventions claimed in '073 patent by any person other than the inventors named in that patent. Accordingly, I have not relied on the inventors' testimony except to the extent that it is corroborated by independent sources.

I have reviewed the documents marked at the depositions of Michael Collins and Gary Thome. None of these documents corroborate any claim that Compaq was in possession of a definite and permanent idea of the complete and operative pre-snoop feature in the TriFlex chipset as it is was ultimately applied in practice to burst multiple cache lines of data. To the contrary, the documents suggest that, if anything, Compaq was still struggling with its pre-snoop feature months after the patent application was filed.

A White Paper prepared by Compaq in November 1994 describes the TriFlex chipset. It states that the initial version of the TriFlex chipset, known as the TriFlex PC was intended to support the EISA bus, and was able to achieve sustained I/O bandwidth of only 33 MBytes/s. (DDX184, HP00264). The same document reports that Compaq's TriFlex PCI chipset was intended to support the PCI bus, and was capable of sustained I/O bandwidth of 123 MBytes/s. (Id). The report documents this performance level for 100/66 MHz processors; that is, processors operating at a clock speed of 100 MHz and communicating over a 66 MHz host bus. (Id). Compaq's corporate representative testified that the 100/66 silicon did not arrive until some time in November. )Thome Dep., 117:4-118:3). Compaq's documents, moreover, indicate that the 66/100 MHz implementation of the TriFlex chipset was still being validated in December 1994, (PDX104, Deposition of Gary Thome, 113:19-115:14, 117:4-118:8), and that the PCI Memory Read Multiple feature -- that is, the ability to burst multiple cache lines of data -- was still being tested as late as January. (PDX103, Thome Dep., 122:7-125:23). The testimony of Compaq's corporate representative was that Compaq would not have released the 100/66 TriFlex chipset -- or any other product -- with this or any other feature until the feature was thoroughly tested, debugged and fully functional. (Thome Dep., 137:7-24, 140:17-141:23). Indeed, Compaq's corporate representative testified that Compaq might well have turned off the pre-snoop feature in the early releases of the TriFlex product if new silicon had been spun for the 100/66 product. (Id., 144:15-145:8). Since that was indeed the case, (PDX104, Thome Dep., 117:4-118:3). Thus, the documentation available strongly suggest that no TriFlex chipset embodied the invention claimed in the '073 Patent until some time in 1995.

The documents which predate the application for the '073 patent are not to the contrary. Compaq meeting minutes for the chipset project explicitly state that the original plan was not to burst more than a single line of data in any transaction, except for reads from write-through memories where cache coherency is essentially a non-issue. (PDX100, 101, and 102; Thome Dep., 92:19-94:6, 94:23-95:11, 98:9-103:15). Subsequent documents do not indicate that multi-line burst reads were actually implemented before the 100/66 chipset. Plaintiff's deposition exhibit 190 is a September 1993 data flow diagram, which shows nothing about whether multi-line bursting was occurring. DDX190. The engineering notebooks of Gary Thome, one of the co-inventors on the '073 patent and Compaq's corporate representative on these issues, likewise contain no corroboration of any earlier multi-line bursting capability. The notebooks do not refer to any testing of the TriFlex's implementation of the PCI "memory read multiple" command, which was the command that would have triggered the multi-line bursting feature. (Thome Dep., 108:24-110:11, 111:24-112:15; PDX192, 193). Nor do they refer to any testing of the chipset that supported the 100/66 processor. PDX192, 193. The only document to mention the "read multiple" feature prior to the October 14 patent application is the daytimer of Michael Collins, another of the '073 co-inventors. (PDX 187). This document was marked at Mr. Collins deposition, and he identified it as, essentially, his lab notebook, but no testimony was taken as to its contents. (Collins Dep., 117:24-119:4). The daytimer indicates that on August 3, 1994, during the debugging process, Compaq ordered the third version of the TriFlex silicon (referred to as CMC-3), Compaq specifically added a "chicken bit" that would allow it to disable the read multiple feature. (PDX187, HP2000687). Subsequent testing, apparently of the CMC-2 silicon indicated the wisdom of this decision, as failures of the read multiple functions, without any indicated fixes, are recorded on August 15 (HP2000710-000711), October 7 (HP2000825), and December 13. (HP2000967). As late as August 17 the question of whether to do read multiples or simply read lines appears to have been an open design issue. (HP2000715).

In sum, the documentation of the TriFlex project does not corroborate any claim by the inventors of the TriFlex chipset to have conceived a clear an definite understanding of how pre-snooping would operate in practice to transfer multiple cache lines of data until, at the earliest, the October 14, 1994 application that ultimately led to the '073 Patent. Rather, the documentation indicates that at the time that patent application was applied, the chipset could not execute a read multiple command, DDX 187, HP2000825 ("PCI multiples turn into rd lines"), and the feature would not be validated until sometime after the start of 1995. Accordingly, I conclude that OPTi's engineers invented the method and apparatus covered by the claims at issue before Compaq did.

## 12. Non-Obviousness

Non-obviousness must be judged by the standard of one of ordinary skill at the time of the invention.

It is helpful to quote from the Tom Shanley deposition with regard to this issue:

116

Q. Okay. Now, I think you testified that the snoop-ahead feature was not a complex concept. Do you recall that testimony?
A. Yes.
Q. That it was fairly simple and straightforward, right?
A. Yes.

117

Q. Now, if it was -- I think you might have also even used the word "obvious" when describing

the snoop-ahead feature. Do you recall that?
A. Simple things usually are. They're obvious when somebody says them.

118

Well, if it was so obvious and such a simple and

119

non-complex concept, why didn't anybody else come out with it before the Triton chipset?
A. I have no idea.
Q. Is it perhaps because there's not enough information in this paragraph of yours on page 286 that didn't allow them to -- without the Intel data sheet, that they couldn't have --
MR. BRIGHAM: Objection; form.
A. To me it's a very simple feature. It's a very simple feature. You know, and as I said earlier, until somebody actually puts something in front of you, a simple solution to something, you know, it may not be obvious until you see it and then you go, well, that's an obvious solution.

Because of the ease with which innovative ideas may appear obvious in hindsight, nonobviousness may be supported by objective indications. In this section I discuss such indications and consider the rejection of such indications in the Colwell Report.

### 12.1. Commercial Success

The measure of commercial success is the extent to which the patent invention is responsible for the commercial success of products incorporating it.

The Colwell report indicates that the OPTi Viper product was not commercially successful. That may be true, but it is not the correct test. The successful products from Intel (Triton) and Compaq (CDC/MDC), as noted by Colwell, incorporated both pre-snooping and prefetching (Colwell, p. 236).

The issue therefore is whether the Intel and Compaq products would have been successful absent the pre-snoop feature. Absent that feature, the chipsets would have had to do PCI reads one at a time - i.e. repeat (snoop, transfer). Considering for example the timing for the Motorola MPC105 (from the Hot Chips presentation), each iteration of that sequence required 16 PCI bus cycles; i.e. PCI bus utilization was 50%, or half of 133MB/sec. This compares to throughput of over 120MB/sec (see Colwell report, p. 237) for the Intel and Compaq products. Notably, the invention disclosure statement that led to Intel's '073 Patent characterized the ability to perform streaming data transfers of the sort made possible by the pre-snoop invention as of strategic importance to Intel and to the future of the PCI standard. (DDX100, at 201NVD00007 and 00006 (pages numbered in reverse order to original sequence)).

In my opinion, a chipset product which did not pre-snoop would not have been commercially successful in competing against a product which did pre-snoop (per the invention of the '906), all other factors being equal.

## 12.2. Others Tried and Failed to Make the Invention

### Amini Patent

As noted above, the device described in the Amini patent did not incorporate the invention of the '906 patent, and in my opinion would have significantly benefited from what the '906 discloses. The inventors were clearly addressing the same or a similar problem.

### IBM Technical Disclosure Bulletin (TDB) Article

As noted above, the device disclosed in the IBM TDB article did not incorporate the invention of the '906 patent, and in my opinion would have significantly benefited from what the '906 discloses. The inventors were clearly addressing the same or a similar problem.

### Hitachi

In a 1993 patent filing (Japanese Patent Office, Official Gazette for Unexamined Patent Alications, Alication Publication No. H7-44460, Filing Date: August 4, 1993, Publication Date: February 14, 1995), Hitachi describes a pre-snoop system in which all of the snoops are performed immediately, at the start of the I/O transfer. The abstract for the patent alication reads as follows:

> In a system wherein a central processing unit equied with cache memory, main storage, bus master, I/O controller and other processing units are connected to a common bus, when a peripheral begins a burst transfer to the main storage, perform snooping of an address region consisting of N consecutive blocks (where N is an arbitrary integer equal to 2 or greater) within the cache memory, including the block region within cache memory corresponding to that access address all at once or in consecutive snoops, thereby achieving efficient access from peripherals by avoiding interruption of the burst transfer due to intermittant snooping.

As may be seen, Hitachi faced the same or a similar problem, and did not make the invention of the '906 patent. The Hitachi design suffers from three disadvantages relative to the '906 design: (a) The number of snooped needed (N) must be known in advance. (b) The snoops are not overlapped with the data transfer, but instead appear to impose a serial delay. (c) There is the problem of writes occurring to the locations to which data is read/written by the peripheral device, either rendering the snoops ineffective or leading to memory inconsistency.

**Motorola MPC105**

As explained above, the Motorola MPC105 did not incorporate the invention of the '906 patent, and in my opinion would have significantly benefited from what the '906 discloses. The inventors were clearly addressing the same or a similar problem.

**Research Literature**

Each of the following research papers/publications was concerned with both prefetching and with cache consistency in a multiprocessor system, but none of them contemplated pre-snoop as an aid for the problems they considered:

Tien-fu Chen, "Data Prefetching for High Performance Processors", Ph.D. Thesis, 1993, U. Washington

Lee, Roland Lun, "The Effectiveness of Caches and Data Prefetch Buffers in Large-Scale Shared Memory Multiprocessors", Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1987.

Frederik Dahlgren, Michel Dubois and Per Stenstrom, "Sequential Hardware Prefetching in Shared-Memory Multiprocessors", IEEETPDS, 6, 7, July, 1995, 773-746.

Dean Tullsen and Susan Eggers, "Limitations of Cache Prefetching on a Bus-Based Multiprocessor," Proc. ISCA 1993, . 278-288.

William Chen, Scott Mahlke, Pohuo Chang and Wen-mei Hwu, "Data Access Microarchitectures for Superscalar Processors with Compiler-Assisted Data Prefetching", Proc. MICRO-24, November, 1991, 69-73.

Alan J Smith, "Cache Memories", ACM Computing Surveys, September, 1982, 473-530.

Jean-Loup Baer and Tien-fu Chen, "An Effective On-Chip Preloading Scheme To Reduce Data Access Penalty", Proc. Supercomputing '91 November, 1991.

Tien-fu Chen and Jean-Loup Baer, "A Performance Study of Software and Hardware Data Prefetching Schemes," Proc. ISCA 21, April, 1994.

Roland Lee, Pen-chung Yew, Duncan Lawrie, "Data Prefetching in Shared Memory Multiprocessor," Proc. IC, August, 1987, . 28-31.

Edward Gornish, Elana Granston, Alexander Veidenbaum, "Compiler-directed Data Prefetching in Multiprocessors with Memory Hierarchies," Proc. ICS, June, 1990, . 354-368.

In short, there is considerable evidence (Motorola, Amini, IBM TDB, Hitachi, various researchers) that others faced the same problem as was addressed by the '906 patent in the same time frame, and failed to make the invention of the '906 patent, which strongly suggests that the invention of the '906 patent was not obvious to ones of extraordinary skill, let alone those of ordinary skill.

## 12.3. The Taking of Licenses Under the Patent by Others

Intel is the largest and most successful maker of high performance microprocessors in the world. That Intel would choose to license the '906 patent suggests to me that Intel believed that the patent was valid and that Intel would like to (or already did) practice the invention of the '906 patent. (Expert Report of Julie L. Davis).

## 12.4. Long Felt Need

As explained by Tom Shanley (Shanley Dep., p. 92-93), there was a need for the pre-snoop invention:

92

A. I don't know. I don't know. To me, when I first saw it, which may have been at this time, it was just common sense. It just made sense.

Q. (By Mr. Brigham) The pre-snoop -- the snoop-ahead feature made sense?

A. Yes.

Q. Why was that?

A. Well, it made sense because performance on the PCI bus was terrible prior to that point in time, prior to this feature, if you will. And it was because the chipsets were designed in a rather primitive manner in that they would always break a PCI bus master's burst transfer as we arrived at every cache line boundary. And, therefore, the biggest transfer you could possibly do in 486-based systems was one 16 byte block of information, four DWords of information.

And for bursting bus masters, like SCSI host bus adapters, devices that needed to transfer large blocks of information very rapidly, performance was terrible. And it was -- when I first saw this feature, it just made sense to me

93

that, you know, you -- if you were in background submitting these snoop-aheads to the processor, then it would allow, assuming we didn't hit on a modified

> line, the burst to continue over cache line boundaries and result in dramatically better performance.

Shanley's characterization of the state of the art is supported by the evidence of Intel's efforts to implement a chipset that could burst across cache lines. This appears to have been on Intel's agenda beginning at least at some point in 1992. Intel was unable to implement such a feature until early 1995. Even after Intel began aggressively designing the feature for the Triton chipset, the Intel engineers' conception of how to implement a workable version of the feature went down a number of dead ends until the final operational device went to market.

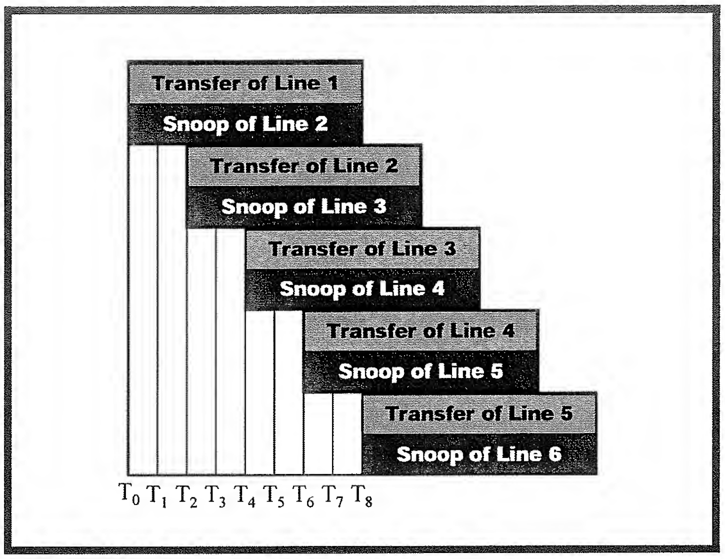
Likewise, Compaq apparently became interested in implementing such a feature in 1992. Its initial work on the TriFlex chipset, however, concluded that bursting would not cross cache lines, and, even after efforts to design a bursting feature were seriously underway, Compaq was unable to implement the concept until early 1995.

## 13. The Asserted Claims are Supported By The Patent's Written Description.

In the Colwell Report, it is argued (248-254) that the '906 patent does not support by written description snooping more than one line ahead. I do not agree with this opinion.

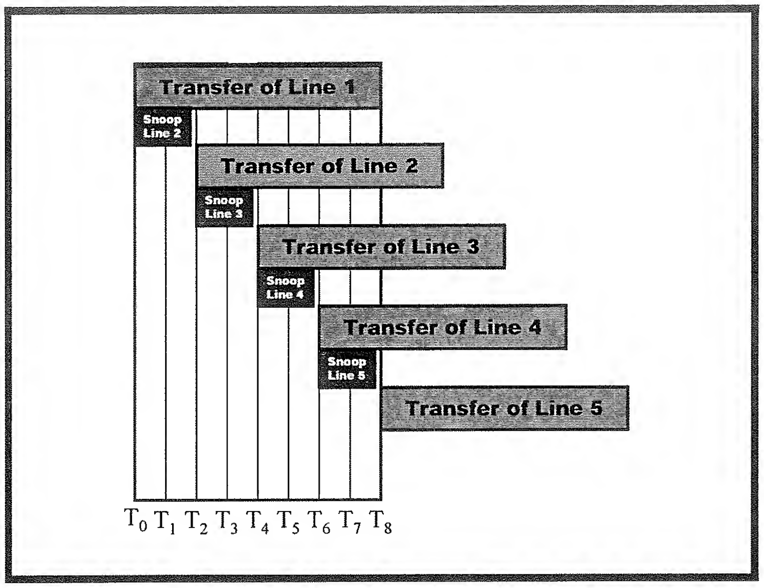
As a preliminary matter, Dr. Colwell's arguments as to the written description requirement seem to be based on a straw man. Dr. Colwell attacks an illustration from a brief submitted by OPTi's counsel as "showing how the claimed pre-snoop method and apparatus works." (Colwell, ¶455). The figure comes from a brief where it was clearly intended merely to illustrate the conceptual point that it is possible to have multiple lines of data in transit at any given time if a buffer is employed in effecting the transfer. This, in turn was intended to make the point that, depending on the timing of the lines' transfer, it is possible that even where each line is snooped exactly once, and even where the snoop of a line occurs during the transfer of the preceding line, it is possible that still earlier lines will still be in transit as well. The passage from the brief is as follows:

> In some chipsets, however, more than one line can be in the process of transferring from memory to a PCI master at a time, because buffers in the chipset are used to store the lines of data during the course of their transfer. In effect, the transfer of such data adds a step to the transfer technique claimed in the Ghosh patents, by creating some additional structure, and imposing on the data a kind of lay-over, during its transfer from the memory to the master. Where multiple lines are in transit, it is certainly possible that a snoop of the "next line" of data which occurs during the transfer of the current line may also occur while a still earlier line is residing in the buffer or otherwise still transferring. Thus, the timing of such transfers and snoops would look something like this:



> What the extent of the overlap would be, obviously depends on the timing of the transfers, but, regardless of what it might include, such implementations of a pre-snoop scheme are well within what those of ordinary skill in the art would have understood to be the scope of the Ghosh invention. (*See* Exhibit G to OPTi's Markman Reply Brief, Declaration Of Alan Jay Smith In Rebuttal To Declaration Of Robert Colwell Of February 14, 2006, Attached as Exhibit F, at ¶¶ 17-18).

Dr. Colwell points out the illustration, which shows that the pre-snoops taking place at any point during the preceding line's transfer, appears to imply that multiple snoops can occur at once, that they can occur after the line to be snooped has started to transfer, and so on. The patent teaches none of this, and in a responsive brief, OPTi agreed that a better illustration of the point being made would be the following, which was provided to the Court and to NVIDIA, but omitted from Dr. Colwell's report:



Dr. Colwell makes no criticism of this corrected illustration, and I am aware of none. [redacted] In addition to not criticizing OPTi's corrected illustration, I note that Dr. Colwell does not suggest that any of the aspects of the first illustrations as to which he is critical are taught by the patent itself. Accordingly, I am of the opinion that none of Dr. Colwell's criticisms of the figure from OPTi's brief identify any impropriety or deficiency in the '906 Patent's disclosure.

Dr. Colwell also argues that "OPTi is attempting to expand its claims to cover snooping more than one line ahead of the line being transferred on the PCI bus." He then explains why such an implementation might be disadvantageous under certain circumstances; namely, where it might result in wasteful snoops. (Colwell Report, ¶¶460-465). He concludes that a person of ordinary skill in the art would not have understood that the system of the invention could be implemented with buffers, since that would entail the need for additional cache coherency mechanisms and would result in unnecessary snoops. (Colwell Report at ¶466).

It is impossible to accurately summarize patent claims; the claims say what the claims say. But here I address what I believe to be the essence of those claims.

The problem addressed by the '906 patent is that the prior art could only perform a long data transfer between a peripheral device on a bus and main memory by breaking up the transfer into, at the most, line-sized increments, with the transfer being delayed each time for the snoop. If the snoop can be completed during the transfer of the previous line, no delay for snoops should occur. The specific embodiment presented in the patent does the snoop one line ahead, and does not prefetch.

My understanding, however, is that the Judge has held in this case that the claims of the patent are not limited to the specific embodiment(s) presented in the patent text. (Markman Order of April 24, 1996, at 2-3. 21-23, 25). The claims for the '906 Patent call for snooping while the previous line is being transferred, and that is what the nVIDIA products do (*See* my Report on Infringement).

Colwell argues that [redacted] prefetching is not contemplated in the '906 patent. My understanding is that a patent can be infringed by a product that incorporates both a patented invention and improvements (possibly patentable) to that invention. [redacted] adds something to the disclosed method of the patent: prefetching. Whether or not this is an improvement to the disclosed invention, the '906 patent presents the best mode known to the inventor for the implementation of his invention. But one of ordinary skill in the art would understand that when additions are made to an invention, the best mode may change. Colwell argues (¶¶ 461-465) that the '906 patent teaches away from the use of pre-snooping more than one line ahead. Given the system contemplated in the '906 patent, snooping one line ahead is optimal. The use of prefetching requires that every prefetched line be snooped, which means that snoops may need to occur more than one line ahead in order to achieve the best performance. This would be understood by one of ordinary skill in the art. [redacted]

I disagree with Dr. Colwell's contention that a person of ordinary skill in the art would have understood the specification to limit the invention to bufferless implementations where no prefetching was performed. The use of prefetch buffering was well known at the time that the application for the '906 Patent was applied for. The use of supplemental cache coherence mechanisms in conjunction with store buffers was well known - see e.g. [Smit82, p.501]. Indeed, the PCI Standard itself suggests that buffers might be used in conjunction with the Memory Read Multiple command. PCI Local Bus Specification, Rev. 2.0, p. 21 (April 30, 1993) ("Memory Read Multiple . . . command is intended to be used . . . when a software transparent buffer is available for storage "). Moreover, multiple chipsets using buffers in conjunction with burst transfers had come to market or been disclosed by the time the '906 Patent had been applied for. Thus, as discussed at length above, Motorola's MPC105 chipset was such a product, as were the chipsets disclosed in the Amini Patent and the IBM Technical disclosure bulletin. In each of these instances, buffering was included as part of a burst transfer mechanism. What these devices lacked was what the '906 Patent added: the insight that the time used for transfer of one line could be used to pre-snoop the next, and that this pre-snooping technique could be used to be used to burst multiple cache lines of data. The Intel (Triton) and Compaq (TriFlex) chipsets had also come to market by this point, and they did just that. Though OPTi had invented the use of pre-snooping to effect multiple line bursting before Intel or Compaq, a person of ordinary skill in the art would read the '906 Patent against what had been achieved using OPTi's invention, and would understand that such implementations were possible. Accordingly, a person of ordinary skill in the art reading the '906 specification at the time the patent was applied for would have understood that the OPTi invention could be used with a buffer.

Accordingly, I believe that the claims are supported by the written description.

Conclusions

For the reasons given above, it is my opinion that asserted claims 1, 7, 8, 9, 21, 26 of OPTi's Patent number 5,710,906 are Valid, despite the opinions presented in the Expert Report of R. Colwell, Ph.D., and I expect to so testify.

Signed this day of May 31, 2006 in Oakland, California.

[illegible]

Alan Jay Smith, Ph.D.